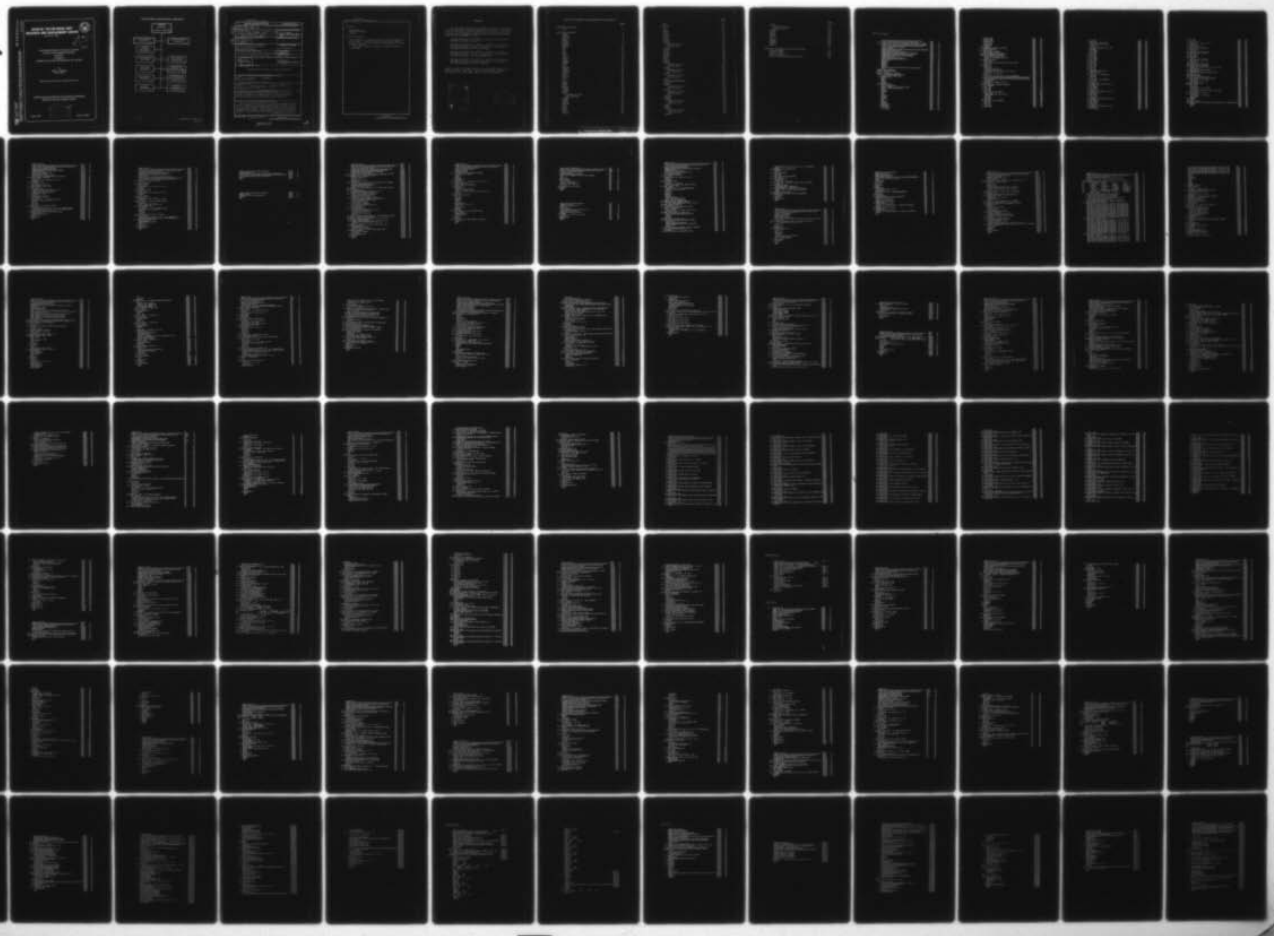
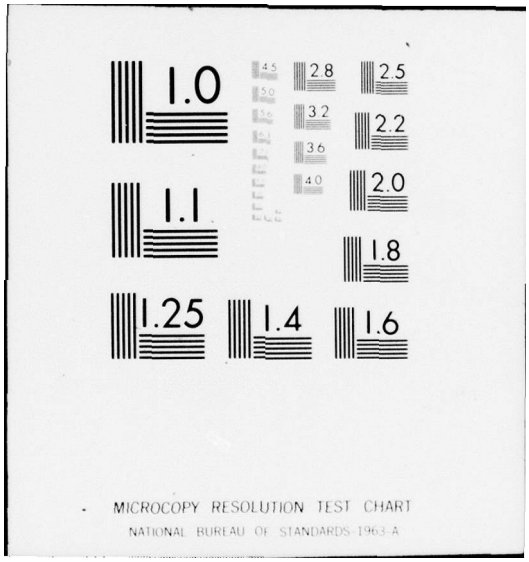
AD-A043 923 DAVID W TAYLOR NAVAL SHIP RESEARCH AND DEVELOPMENT CE--ETC F/G 9/2
MAINTENANCE MANUAL FOR AUDIT. A SYSTEM FOR ANALYZING SESCOMP SO--ETC(U)
AUG 77 R J WYBRANIEC, R REGEN

UNCLASSIFIED DTNSRDC-77-0075-VOL-2 NL

1 OF 2
ADA043923




1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

Report 77-0075

MAINTENANCE MANUAL FOR AUDIT, A SYSTEM FOR ANALYZING SESCOMP SOFTWARE
VOLUME 2 – APPENDIX B – LISTINGS OF THE AUDIT SOFTWARE FOR THE CDC 6000

DAVID W. TAYLOR NAVAL SHIP RESEARCH AND DEVELOPMENT CENTER

Bethesda, Md. 20084

MAINTENANCE MANUAL FOR AUDIT, A SYSTEM FOR ANALYZING SESCOMP SOFTWARE
VOLUME 2
APPENDIX B
LISTINGS OF THE AUDIT SOFTWARE FOR THE CDC 6000

by

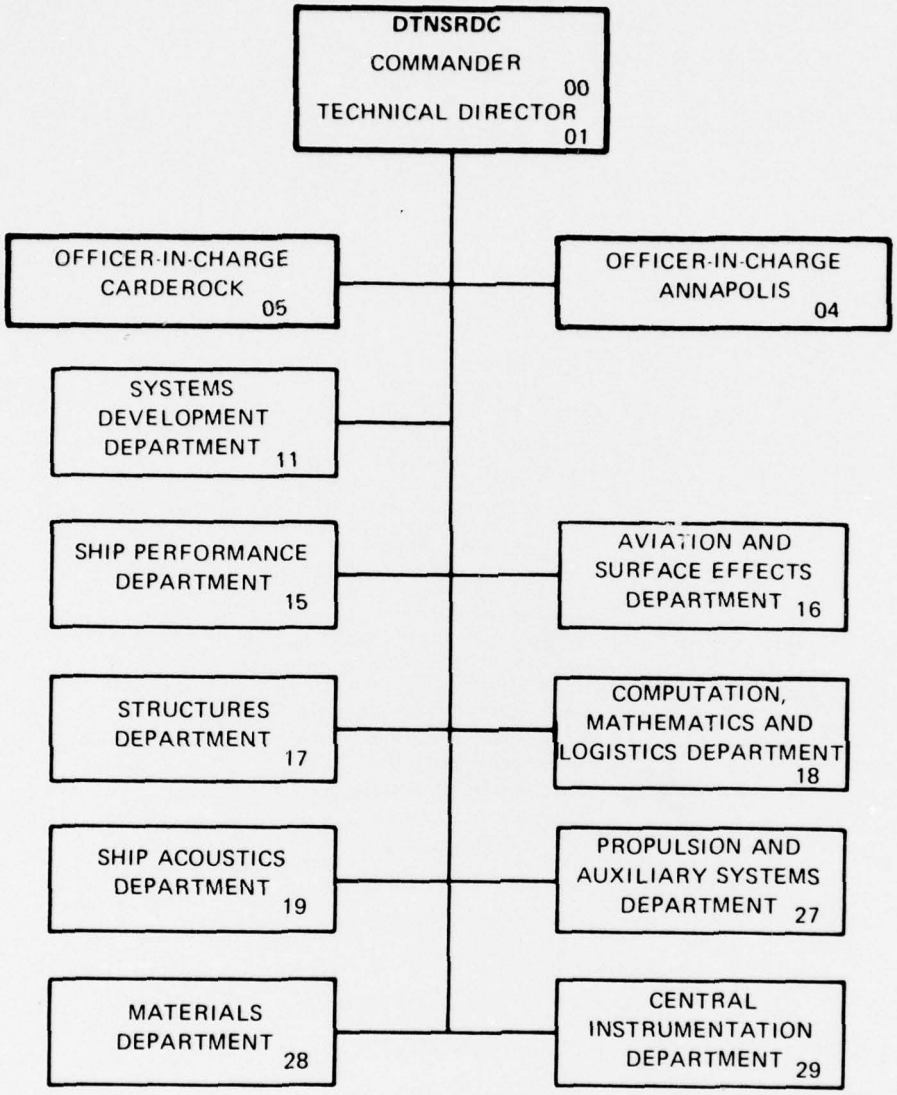
Robert J. Wybraniec
Richard Regen

COMPUTATION, MATHEMATICS, AND LOGISTICS DEPARTMENT
RESEARCH AND DEVELOPMENT REPORT

August 1977

Report 77-0

MAJOR DTNSRDC ORGANIZATIONAL COMPONENTS



GPO 905-678

NDW-DTNSRDC 3960/43b (Rev. 11-75)
GPO 928-10

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

(14) 1. REPORT NUMBER: DTNSRDC-77-0075-Vol-2

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

(6) 4. TITLE (and Subtitle): MAINTENANCE MANUAL FOR AUDIT, A SYSTEM FOR ANALYZING SESCOMP SOFTWARE, VOLUME 2: APPENDIX B • LISTINGS OF THE AUDIT SOFTWARE FOR THE CDC 6000.

(9) 5. TYPE OF REPORT & PERIOD COVERED: Final rept.

6. PERFORMING ORG. REPORT NUMBER

(10) 7. AUTHOR(s): Robert J. Wybraniec, Richard Regen

8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS: David W. Taylor Naval Ship Research and Development Center, Bethesda, Maryland 20084

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: (See reverse side)

11. CONTROLLING OFFICE NAME AND ADDRESS: Navy Surface Effect Ships Project (PMS 304), P.O. Box 34401 - Bethesda, Maryland 20084

(11) 12. REPORT DATE: August 1977

13. NUMBER OF PAGES: 188

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): (12) 186 p.

15. SECURITY CLASS. (of this report): UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

APPROVED FOR PUBLIC RELEASE: DISTRIBUTION UNLIMITED

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

(16) SSH15, S0308

18. SUPPLEMENTARY NOTES

(17) SSH15001, S0308001

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

SESCOMP, SESCOMPSPEC's, Software Verification, Software Engineering, Reliability, Graph Theory, FORTRAN Software, Modules, Flow Analysis, Variable Precision Execution, Parser, Roll Call, Portability

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The AUDIT documentation provides the maintenance programmer personnel with the information to effectively maintain and use the AUDIT software. The AUDIT software examines FORTRAN computer programs or modules developed under the sescomp system for compliance with certain prescribed standards (SESCOMPSPEC's) and produces reports detailing the deviations from those standards. The AUDIT software also examines a program unit to detect and

(Continued on reverse side)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE S/N 0102-LF-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

387682

/B

(Block 10)

63534N, 19588,
SSH15001 and S0308001,
11837001

(Block 20 continued)

report improper use of undefined variables along the program unit's possible paths. In addition, AUDIT has an option which enables the user to test the effect of changes in word length on the output of computer programs.

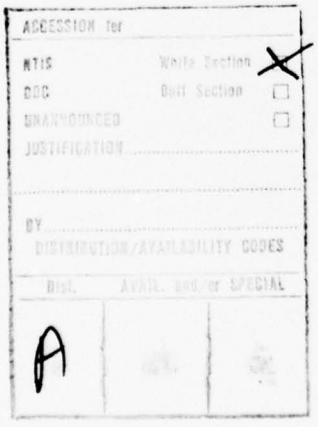
This report contains the listings of the AUDIT software for the CDC 6000.

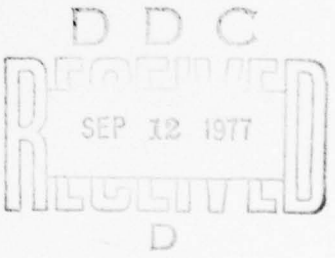
FOREWORD

The use and maintenance of AUDIT, a software system for analyzing SESCOMP contractor-supplied software, is documented as a set of four separately bound David W. Taylor Naval Ship Research and Development Center volumes sharing the common report number--DTNSRDC 77-0075:

- Maintenance Manual for AUDIT, a System for Analyzing SESCOMP Software, Volume 1
- Maintenance Manual for AUDIT, a System for Analyzing SESCOMP Software, Volume 2; Appendix B - Listings of the AUDIT Software for the CDC 6000
- Maintenance Manual for AUDIT, a System for Analyzing SESCOMP Software, Volume 3; Appendix C - Listings of the AUDIT Software for the UNIVAC 1108
- Maintenance Manual for AUDIT, a System for Analyzing SESCOMP Software, Volume 4; Appendix D - Listings of the AUDIT Software for the IBM 360

Volume 1 describes AUDIT and the use and maintenance of the AUDIT software. The other three volumes offer software listings for the CDC 6000, UNIVAC 1108, and IBM 360.





INDEX TO PROGRAMS AND SUBPROGRAMS IN APPENDIX B

AUDIT Main Program

```
      PROGRAM MAIN(INPUT=65,OUTPUT=65,TAPE5=INPUT,TAPE6=OUTPUT,          MAIN      3
     * TAPE7=65,TAPE8=65,TAPE9=513,TAPE4=513,TAPE19=513)                 MAIN      4
C****IF UNIVAC 1108 - PLACE A "C" IN COLUMN 1 OF PREVIOUS TWO CARDS      MAIN      5
      COMMON A(1326),D(500),IDTBL(8,500),INITIO(3),LASTIO(3),ISRCH(3),   RICH      2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,         CY58A    60
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                   RICH      4
      DIMENSION IORD(15)                                                 MAIN      7
      COMMON/GLOBAL/NBLK,NREF,NSUBS,BLKTBL(200),EXTTBL(100),ISUBS(100)   MAIN      8
      COMMON/INPOUT/NCALL,IN,IOP                                         MAIN      9
      COMMON/LABELS/STATRA(2,200),NLABEL                                 MAIN     10
      COMMON/LIST/NLIST,NINTFC,ISUBLT(2,200),INTFAC(300)                 MAIN     11
      COMMON/DOLOOP/ISTACK(4,50),NSTACK,ILOOP,IOVFLW                     MAIN     12
      COMMON/BASBLK/IBLOCK(2500),NBLOCK,NB,NBRNCH                        CY58A     1
      COMMON/FLOW/IFL,IRP                                                MAIN     14
      COMMON/STFUNC/NSTFNC,ISTFNC(10)                                    MAIN     15
      INTEGER A,BLANK,STATRA,BLKTBL,EXTTBL                               MAIN     16
      DATA BLANK/1H /                                                    MAIN     17
      REWIND 4                                                           MAIN     18
      REWIND 19                                                          MAIN     19
      REWIND 7                                                           MAIN     20
      REWIND 8                                                           MAIN     21
      REWIND 9                                                           MAIN     22
      DATA IORD/29,30,31,32,25,26,19,20,21,22,23,24,26,27,35/            MAIN     23
      IOP=8                                                              MAIN     24
C**READ INPUT PARAMETERS                                                 MAIN     25
C**MODE - MODE OF OPERATION                                              MAIN     26
C**IN - INPUT FILE                                                       MAIN     27
C**IFL - FLOW ANALYSIS PARAMETER                                         MAIN     28
C**INTR - INTRINSIC FUNCTION PARAMETER                                   MAIN     29
      READ(5,12) MODE,IN,IFL,INTR                                        MAIN     30
   12 FORMAT(I1,3I4)                                                     MAIN     31
      NCALL=0                                                            MAIN     32
      NREF=0                                                             MAIN     33
      NBLK=0                                                             MAIN     34
      NSUBS=0                                                            MAIN     35
      IRP=IFL-1                                                          MAIN     36
C** READ SESCOMP LIST                                                    MAIN     37
      READ(4) NLIST,NINTFC                                               MAIN     38
      READ(4) ((ISUBLT(I,J),I=1,2),J=1,NLIST)                            MAIN     39
      READ(4) (INTFAC(I),I=1,NINTFC)                                     MAIN     40
    4 ISTAT=0                                                            MAIN     41
      IOVFLW=0                                                           MAIN     42
      NLABEL=0                                                           MAIN     43
      NID=0                                                              MAIN     44
      JJ=0                                                               MAIN     45
      IBLKDT=0                                                           MAIN     46
      NSTACK=0                                                           MAIN     47
      NBLOCK=0                                                           MAIN     48
      ILOOP=0                                                            MAIN     49
      NB=0                                                               MAIN     50
      IBLKST=0                                                           MAIN     51
      NSTFNC=0                                                           MAIN     52
      WRITE(6,13)                                                        MAIN     53
   13 FORMAT(1H1)                                                        MAIN     54
      IFNCNM=BLANK                                                       MAIN     55
      DO 2 I=1,8                                                         MAIN     56
      DO 2 J=1,500                                                       MAIN     57
```

```
    2 IOTBL(I,J)=0                                                    MAIN      58
      DO 3 I=1,200                                                    MAIN      59
      STATRA(1,I)=0                                                   MAIN      60
    3 STATRA(2,I)=0                                                   MAIN      61
      DO 5 I=1,3                                                      MAIN      62
      INITID(I)=0                                                     MAIN      63
    5 LASTID(I)=0                                                     MAIN      64
  700 CONTINUE                                                        MAIN      65
      IF(NBLOCK .GT. 2500) GO TO 7000                                 MAIN      66
C** READ NEXT STATEMENT                                               MAIN      67
      CALL BUILD                                                      MAIN      68
C** WRITE OUT NEXT STATEMENT                                          MAIN      69
      WRITE(6,1000) (A(I),I=1,N)                                      MAIN      70
 1000 FORMAT(//6X,100A1,13(/6X,100A1))                                MAIN      71
C** SKIP IF COMMENT OR BLANK CARD                                     MAIN      72
      IF(A(1) .EQ. 1HC) GO TO 700                                     MAIN      73
      IF(NEXT(1) .EQ. BLANK) GO TO 700                                MAIN      74
      JJ=JJ+1                                                         MAIN      75
C** CLASSIFY STATEMENT                                                MAIN      76
      CALL CLASS                                                      MAIN      77
      IF(ITYP .GT. 18 .AND. ITYP .NE. 28) GO TO 1320                  MAIN      78
      CALL STATNO                                                     MAIN      79
 1320 JPTR=7                                                          MAIN      80
      IPREC=ISTAT                                                     MAIN      81
      ISTAT=ITYP                                                      MAIN      82
      IF(JJ .EQ. 1) GO TO 7                                           MAIN      83
      IF(IBLKOT .EQ. 0) GO TO 6                                       MAIN      84
      IF(ITYP .GE. 18 .AND. ITYP .LE. 27) GO TO 8                     MAIN      85
      GO TO 220                                                       MAIN      86
    6 IF(LTYP .EQ. 9) GO TO 120                                       MAIN      87
      GO TO 8                                                         MAIN      88
    7 IF(ITYP .LT. 29 .OR. ITYP .GT. 32) CALL ERROR(2)                MAIN      89
    8 GO TO (60,70,80,90,100,110,20,140,20,20,40,40,50,50,50,120,130, MAIN      90
     1 2000,10,10,10,10,10,170,180,190,200,210,150,30,30,160,220,220, MAIN      91
     2 220,220),ITYP                                                  MAIN      92
   10 DO 15 I=1,11                                                    MAIN      93
      IF(IPREC .EQ. IORD(I)) GO TO 17                                 MAIN      94
   15 CONTINUE                                                        MAIN      95
C** GO TO APPROPRIATE STATEMENT PROCESSOR                             MAIN      96
      CALL ERROR(2)                                                   MAIN      97
   17 CALL TYPE                                                       MAIN      98
      GO TO 500                                                       MAIN      99
   20 CALL SIMP                                                       MAIN     100
      GO TO 500                                                       MAIN     101
   30 IF(JJ .NE. 1) CALL ERROR(2)                                     MAIN     102
      DO 35 I=1,20                                                    MAIN     103
      IF(NEXT(JPTR) .NE. 1HF) GO TO 35                                MAIN     104
      JPTR=JPTR-1                                                     MAIN     105
      CALL SUB                                                        MAIN     106
      GO TO 500                                                       MAIN     107
   35 CONTINUE                                                        MAIN     108
   38 IF(JJ .NE. 1) CALL ERROR(2)                                     MAIN     109
      CALL SUB                                                        MAIN     110
      GO TO 500                                                       MAIN     111
   40 CALL IO                                                         MAIN     112
      GO TO 500                                                       MAIN     113
   50 CALL AUXIO                                                      MAIN     114
```

```
      GO TO 500                                                          MAIN      115
   60 CALL INIT                                                          MAIN      116
      WRITE(6,66) (A(I),I=1,N)                                           MAIN      117
      IF(ITYP .NE. 35) GO TO 500                                         MAIN      118
      ISTAT=35                                                           MAIN      119
      DO 67 I=1,15                                                       MAIN      120
      IF(IPREC .EQ. IORD(I)) GO TO 500                                   MAIN      121
   67 CONTINUE                                                           MAIN      122
      CALL ERROR(2)                                                      MAIN      123
      GO TO 500                                                          MAIN      124
   70 CALL ASSIGN                                                        MAIN      125
      GO TO 500                                                          MAIN      126
   80 CALL GOTO                                                          MAIN      127
      GO TO 500                                                          MAIN      128
   90 CALL ASGOTO                                                        MAIN      129
      GO TO 500                                                          MAIN      130
  100 CALL CTGOTO                                                        MAIN      131
      GO TO 500                                                          MAIN      132
  110 CALL ARIF                                                          MAIN      133
      GO TO 500                                                          MAIN      134
  120 CALL LOGIF                                                         MAIN      135
      GO TO 500                                                          MAIN      136
  130 CALL DO                                                            MAIN      137
      GO TO 500                                                          MAIN      138
  140 CALL CALL                                                          MAIN      139
      WRITE(6,66) (A(I),I=1,N)                                           MAIN      140
   66 FORMAT(6X,72A1)                                                    MAIN      141
      GO TO 500                                                          MAIN      142
  150 IF(JJ .NE. 1) CALL ERROR(2)                                        MAIN      143
      IBLKDT=1                                                           MAIN      144
      CALL SIMP                                                          MAIN      145
      GO TO 500                                                          MAIN      146
  160 IF(JJ .NE. 1) CALL ERROR(2)                                        MAIN      147
      CALL PROG                                                          MAIN      148
      GO TO 500                                                          MAIN      149
  170 DO 175 I=1,12                                                      MAIN      150
      IF(IPREC .EQ. IORD(I)) GO TO 177                                   MAIN      151
  175 CONTINUE                                                           MAIN      152
      CALL ERROR(2)                                                      MAIN      153
  177 CALL DIMEN                                                         MAIN      154
      GO TO 500                                                          MAIN      155
  180 DO 185 I=1,5                                                       MAIN      156
      IF(IPREC .EQ. IORD(I)) GO TO 187                                   MAIN      157
  185 CONTINUE                                                           MAIN      158
      CALL ERROR(2)                                                      MAIN      159
  187 CALL COM                                                           MAIN      160
      GO TO 500                                                          MAIN      161
  190 DO 195 I=1,13                                                      MAIN      162
      IF(IPREC .EQ. IORD(I)) GO TO 197                                   MAIN      163
  195 CONTINUE                                                           MAIN      164
      CALL ERROR(2)                                                      MAIN      165
  197 CALL EQUIV                                                         MAIN      166
      GO TO 500                                                          MAIN      167
  200 DO 205 I=1,14                                                      MAIN      168
      IF(IPREC .EQ. IORD(I)) GO TO 207                                   MAIN      169
  205 CONTINUE                                                           MAIN      170
      CALL ERROR(2)                                                      MAIN      171
```

```
  207 CALL DATA                                                          MAIN      172
      GO TO 500                                                          MAIN      173
  210 DO 215 I=1,6                                                       MAIN      174
      IF(IPREC .EQ. IORD(I)) GO TO 217                                   MAIN      175
  215 CONTINUE                                                           MAIN      176
      CALL ERROR(2)                                                      MAIN      177
  217 IF(N .GT. 72) GO TO 240                                            MAIN      178
      WRITE(IOP,218) (A(I),I=1,N)                                        MAIN      179
  218 FORMAT(72A1)                                                       MAIN      180
      GO TO 250                                                          MAIN      181
  240 WRITE(IOP,245) (A(I),I=1,N)                                        MAIN      182
  245 FORMAT(72A1/(5X,1H*,66A1))                                         MAIN      183
  250 CALL FRMAT                                                         MAIN      184
      GO TO 700                                                          MAIN      185
  220 CALL ERROR(1)                                                      MAIN      186
  500 CONTINUE                                                           MAIN      187
      IF(N .GT. 72) GO TO 540                                            MAIN      188
      WRITE(IOP,520) (A(I),I=1,N)                                        MAIN      189
  520 FORMAT(72A1)                                                       MAIN      190
      GO TO 600                                                          MAIN      191
  540 WRITE(IOP,545) (A(I),I=1,N)                                        MAIN      192
  545 FORMAT(72A1/(5X,1H*,66A1))                                         MAIN      193
  600 IF(MODE .NE. 1) GO TO 700                                          MAIN      194
      IF(ITYP .LT. 30 .OR. ITYP .GT. 32) GO TO 700                       MAIN      195
      WRITE(IOP,610)                                                     MAIN      196
  610 FORMAT(5X,15H COMPLEX Q1COMP)                                      MAIN      197
      WRITE(IOP,620)                                                     MAIN      198
  620 FORMAT(5X,24H DOUBLE PRECISION Q1DPRE)                             MAIN      199
      GO TO 700                                                          MAIN      200
 2000 WRITE(IOP,2020)                                                    MAIN      201
 2020 FORMAT(6X,3HEND)                                                   MAIN      202
      IF(N .NE. 72) WRITE(6,2100)                                        MAIN      203
 2100 FORMAT(6X,22H ILLEGAL END STATEMENT)                               MAIN      204
      CALL SUBCHK                                                        MAIN      205
      IF(INTR .NE. 1) GO TO 3205                                         MAIN      206
      CALL INTRIN                                                        MAIN      207
C** DISPLAY SYMBOL TABLE                                                 MAIN      208
 3205 CALL SYMTAB                                                        MAIN      209
      CALL GRT                                                           MAIN      210
      CALL COMCHK                                                        MAIN      211
      IF(IOVFLW .EQ. 1) GO TO 3210                                       MAIN      212
      CALL LOOPCK                                                        MAIN      213
      IF(IFL .EQ. 0 .OR. IBLKDT .EQ. 1) GO TO 3210                       MAIN      214
      CALL FLOWCK                                                        MAIN      215
 3210 IF(IERR .NE. 2) GO TO 4                                            MAIN      216
      CALL GLOTAB                                                        MAIN      217
      IF(MODE .EQ. 1) GO TO 6000                                         MAIN      218
      CALL GENROL                                                        MAIN      219
      REWIND 9                                                           MAIN      220
 6000 REWIND 8                                                           MAIN      221
      STOP                                                               MAIN      222
 7000 WRITE(6,7005)                                                      MAIN      223
 7005 FORMAT(/////5X,54H OVERFLOW OF BASIC BLOCK TABLE - PROCESSING TERM MAIN      224
     *INATED)                                                            MAIN      225
      STOP                                                               MAIN      226
      END                                                                MAIN      227
```

```
      SUBROUTINE ARIF                                                    ARIF     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),   RICH     2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,         CY58A   80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                   RICH     4
      COMMON/TYP/NQQ,RHSTYP,NQ2,NQ3,LHSTYP                               ARIF     4
      COMMON/STRING/NTYPE,NSTR,STR(500)                                  ARIF     5
      COMMON/LABELS/STATRA(2,200),NLABEL                                 ARIF     6
      COMMON/BASBLK/IBLOCK(2500),NBLOCK,NB,NBRNCH                        CY58A   32
      INTEGER A,STATRA,STR,COMMA,BLANK,RHSTYP,AY,EF                      ARIF     8
      INTEGER BITPUT                                                     ARIF     9
      DATA LPAR/1H(/,COMMA/1H,/,BLANK/1H /,AY/1HI/,EF/1HF/               ARIF    10
C** ARITHMETIC IF STATEMENT PROCESSOR                                    ARIF    11
      IF(NEXT(JPTR) .NE. AY) GO TO 20                                    ARIF    12
      IF(NEXT(JPTR) .NE. EF) GO TO 20                                    ARIF    13
      IF(NEXT(JPTR) .NE. LPAR) GO TO 20                                  ARIF    14
      JPTR=JPTR-1                                                        ARIF    15
C** PARSE THE EXPRESSION                                                 ARIF    16
      CALL EXPR                                                          ARIF    17
      NSTR=NSTR+1                                                        ARIF    18
      STR(NSTR)= -5                                                      ARIF    19
      NTYPE=1                                                            ARIF    20
      CALL PARSE                                                         ARIF    21
C** PROCESS FUNCTION REFERENCES                                          ARIF    22
      CALL FNCSTR                                                        ARIF    23
      IF(RHSTYP .EQ. 1) CALL ERROR(42)                                   ARIF    24
C** STORE BASIC BLOCKS                                                   ARIF    25
      CALL BLKSTR                                                        ARIF    26
      NBRNCH=0                                                           ARIF    27
      DO 10 I=1,3                                                        ARIF    28
C** GET NEXT BRANCH                                                      ARIF    29
      CALL GNLE                                                          ARIF    30
      IF(JTYP .NE. 5) GO TO 20                                           ARIF    31
C** GET STATEMENT NUMBER TABLE LOCATION AND SET "REFERENCED" FLAG        ARIF    32
      CALL STSRCH                                                        ARIF    33
      STATRA(2,LOC)=BITPUT(STATRA(2,LOC),1,12)                           ARIF    34
      IF( NBRNCH .EQ. 0) GO TO 5                                         ARIF    35
C** CHECK FOR DUPLICATE BRANCHES                                         ARIF    36
      DO 3 J=1,NBRNCH                                                    ARIF    37
      IF(LOC .EQ. IBLOCK(NBLOCK-J+1)) GO TO 7                            ARIF    38
    3 CONTINUE                                                           ARIF    39
C** STORE BRANCH IN BASIC BLOCK TABLE                                    ARIF    40
    5 NBLOCK=NBLOCK+1                                                    ARIF    41
      IBLOCK(NBLOCK)=LOC                                                 ARIF    42
C** INCREMENT BRANCH COUNTER                                             ARIF    43
      NBRNCH=NBRNCH+1                                                    ARIF    44
    7 IF(I .EQ. 3) GO TO 10                                              ARIF    45
      IF(NEXT(JPTR) .NE. COMMA) GO TO 20                                 ARIF    46
   10 CONTINUE                                                           ARIF    47
      IF(NEXT(JPTR) .NE. BLANK) GO TO 20                                 ARIF    48
      NB=1                                                               ARIF    49
      RETURN                                                             ARIF    50
   20 CALL ERROR(7)                                                      ARIF    51
      RETURN                                                             ARIF    52
      END                                                                ARIF    53
```

```
      SUBROUTINE ASGOTO                                                 ASGOTO     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH       2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A     80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH       4
      COMMON/LABELS/STATRA(2,200),NLABEL                                ASGOTO     4
      COMMON/BASBLK/IBLOCK(2500),NBLOCK,NB,NBRNCH                       CY58A     29
      DIMENSION IALPH(4)                                                ASGOTO     6
      INTEGER STATRA,BLANK,COMMA,RPAR,A                                 ASGOTO     7
      INTEGER BITPUT,BITGET                                             ASGOTO     8
      DATA (IALPH(I),I=1,4)/1HG,1HO,1HT,1HO/                            ASGOTO     9
      DATA BLANK/1H /,COMMA/1H,/,LPAR/1H(/,RPAR/1H)/                    ASGOTO    10
C** ASSIGNED GO TO STATEMENT PROCESSOR                                  ASGOTO    11
      DO 5 I=1,4                                                        ASGOTO    12
      IF(NEXT(JPTR) .NE. IALPH(I)) GO TO 30                             ASGOTO    13
    5 CONTINUE                                                          ASGOTO    14
C** GET VARIABLE REFERENCE                                              ASGOTO    15
      CALL GNLE                                                         ASGOTO    16
      IF(JTYP .NE. 2) GO TO 30                                          ASGOTO    17
C** GET SYMBOL TABLE LOCATION                                           ASGOTO    18
      CALL SEARCH                                                       ASGOTO    19
      IF(ISRCH(2) .EQ. 1) CALL ERROR(10,NXTID)                          ASGOTO    20
      IF(ISRCH(1) .EQ. 1) GO TO 10                                      ASGOTO    21
      IDTYP=1                                                           ASGOTO    22
      CALL STORE                                                        ASGOTO    23
      LOC=NID                                                           ASGOTO    24
C** GET TYPE AND CHECK THAT IT IS INTEGER VARIABLE                      ASGOTO    25
   10 CALL IMPTYP                                                       ASGOTO    26
      IF(BITGET(IDTBL(3,LOC),10,3) .NE. 4) CALL ERROR(39,NXTID)         ASGOTO    27
      IF(BITGET(IDTBL(3,LOC),1,1) .EQ. 1) CALL ERROR(14,NXTID)          ASGOTO    28
      IF(NEXT(JPTR) .NE. COMMA) GO TO 30                                ASGOTO    29
      IF(NEXT(JPTR) .NE. LPAR) GO TO 30                                 ASGOTO    30
C** STORE REFERENCE IN BASIC BLOCK TABLE                                ASGOTO    31
      NBLOCK=NBLOCK+1                                                   ASGOTO    32
      IBLOCK(NBLOCK)=5000+LOC                                           ASGOTO    33
      NBRNCH=0                                                          ASGOTO    34
C** GET NEXT BRANCH                                                     ASGOTO    35
   20 CALL GNLE                                                         ASGOTO    36
      IF(JTYP .NE. 5) GO TO 30                                          ASGOTO    37
C** GET STATEMENT NUMBER TABLE LOCATION AND SET "GOTO" FLAG             ASGOTO    38
      CALL STSRCH                                                       ASGOTO    39
      STATRA(2,LOC)=BITPUT(STATRA(2,LOC),1,12)                          ASGOTO    40
      IF(NBRNCH .EQ. 0) GO TO 25                                        ASGOTO    41
C** CHECK FOR DUPLICATE BRANCHES                                        ASGOTO    42
      DO 22 I=1,NBRNCH                                                  ASGOTO    43
      IF(LOC .EQ. IBLOCK(NBLOCK-I+1)) GO TO 27                          ASGOTO    44
   22 CONTINUE                                                          ASGOTO    45
C** STORE BRANCH IN BASIC BLOCK TABLE                                   ASGOTO    46
   25 NBLOCK=NBLOCK+1                                                   ASGOTO    47
      IBLOCK(NBLOCK)=LOC                                                ASGOTO    48
C** INCREMENT BRANCH COUNTER                                            ASGOTO    49
      NBRNCH=NBRNCH+1                                                   ASGOTO    50
   27 IF(NEXT(JPTR) .EQ. COMMA) GO TO 20                                ASGOTO    51
      IF(A(JPTR-1) .NE. RPAR) GO TO 30                                  ASGOTO    52
      IF(NEXT(JPTR) .NE. BLANK) GO TO 30                                ASGOTO    53
      NB=1                                                              ASGOTO    54
      RETURN                                                            ASGOTO    55
   30 CALL ERROR(7)                                                     ASGOTO    56
      RETURN                                                            ASGOTO    57
      END                                                               ASGOTO    58
```

```
      SUBROUTINE ASSIGN                                                 ASSIGN     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH       2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A      80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH       4
      COMMON/LABELS/STATRA(2,200),NLABEL                                ASSIGN     4
      COMMON/BASBLK/IBLOCK(2500),NBLOCK,NB,NBRNCH                       CY58A      30
      DIMENSION IALPH(6)                                                ASSIGN     6
      INTEGER BLANK,TEE,OH,STATRA                                       ASSIGN     7
      INTEGER BITPUT,BITGET                                             ASSIGN     8
      DATA BLANK/1H /,TEE/1HT/,OH/1HO/                                  ASSIGN     9
      DATA (IALPH(I),I=1,6)/1HA,1HS,1HS,1HI,1HG,1HN/                    ASSIGN    10
C** ASSIGN STATEMENT PROCESSOR                                          ASSIGN    11
      DO 5 I=1,6                                                        ASSIGN    12
      IF(NEXT(JPTR) .NE. IALPH(I)) GO TO 20                             ASSIGN    13
    5 CONTINUE                                                          ASSIGN    14
C** GET STATEMENT LABEL                                                 ASSIGN    15
      CALL GNLE                                                         ASSIGN    16
      IF(JTYP .NE. 5) GO TO 20                                          ASSIGN    17
C** SEARCH STATEMENT NUMBER TABLE                                       ASSIGN    18
      CALL STSRCH                                                       ASSIGN    19
C** SET "ASSIGN" FLAG                                                   ASSIGN    20
      STATRA(2,LOC)=BITPUT(STATRA(2,LOC),1,12)                          ASSIGN    21
      IF(NEXT(JPTR) .NE. TEE) GOTO 20                                   ASSIGN    22
      IF(NEXT(JPTR) .NE. OH) GO TO 20                                   ASSIGN    23
C** GET VARIABLE REFERENCE                                              ASSIGN    24
      CALL GNLE                                                         ASSIGN    25
      IF(JTYP .NE. 2) GO TO 20                                          ASSIGN    26
C** GET SYMBOL TABLE LOCATION                                           ASSIGN    27
      CALL SEARCH                                                       ASSIGN    28
      IF(ISRCH(2) .EQ. 1) CALL ERROR(10,NXTID)                          ASSIGN    29
      IF(ISRCH(1) .EQ. 1) GO TO 10                                      ASSIGN    30
      IDTYP=1                                                           ASSIGN    31
      CALL STORE                                                        ASSIGN    32
      LOC=NID                                                           ASSIGN    33
C** CHECK THAT IT IS AN INTEGER VARIABLE                                ASSIGN    34
   10 CALL IMPTYP                                                       ASSIGN    35
      IF(BITGET(IDTBL(3,LOC),10,3) .NE. 4) CALL ERROR(39,NXTID)         ASSIGN    36
      IF(BITGET(IDTBL(3,LOC),1,1) .EQ. 1) CALL ERROR(14,NXTID)          ASSIGN    37
      IF(NEXT(JPTR) .NE. BLANK) GO TO 20                                ASSIGN    38
C** STORE ASSIGNED VARIABLE IN BASIC BLOCK TABLE                        ASSIGN    39
      NBLOCK=NBLOCK+1                                                   ASSIGN    40
      IBLOCK(NBLOCK)=4000+LOC                                           ASSIGN    41
      RETURN                                                            ASSIGN    42
   20 CALL ERROR(7)                                                     ASSIGN    43
      RETURN                                                            ASSIGN    44
      END                                                               ASSIGN    45
```

```
      SUBROUTINE AUXIO                                                    AUXIO     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),    RICH      2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,          CY58A    80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                    RICH      4
      COMMON/BASBLK/IBLOCK(2500),NBLOCK,NB,NBRNCH                         CY58A    12
      DIMENSION IALPH1(6),IALPH2(9),IALPH3(7)                             AUXIO     5
      INTEGER BITGET                                                      AUXIO     6
      DATA (IALPH1(I),I=1,6)/1HR,1HE,1HW,1HI,1HN,1HD/                     AUXIO     7
      DATA (IALPH2(I),I=1,9)/1HB,1HA,1HC,1HK,1HS,1HP,1HA,1HC,1HE/         AUXIO     8
      DATA (IALPH3(I),I=1,7)/1HE,1HN,1HD,1HF,1HI,1HL,1HE/                 AUXIO     9
C** AUXILARY I/O STATEMENT PROCESSOR                                      AUXIO    10
      IT=16-ITYP                                                          AUXIO    11
      IF(IT-2) 25,15,5                                                    AUXIO    12
C** REWIND STATEMENT                                                      AUXIO    13
    5 DO 10 I=1,6                                                         AUXIO    14
      IF(NEXT(JPTR) .NE. IALPH1(I)) GO TO 50                              AUXIO    15
   10 CONTINUE                                                            AUXIO    16
      GO TO 40                                                            AUXIO    17
C** BACKSPACE STATEMENT                                                   AUXIO    18
   15 DO 20 I=1,9                                                         AUXIO    19
      IF(NEXT(JPTR) .NE. IALPH2(I)) GO TO 50                              AUXIO    20
   20 CONTINUE                                                            AUXIO    21
      GO TO 40                                                            AUXIO    22
C** ENDFILE STATEMENT                                                     AUXIO    23
   25 DO 30 I=1,7                                                         AUXIO    24
      IF(NEXT(JPTR) .NE. IALPH3(I)) GO TO 50                              AUXIO    25
   30 CONTINUE                                                            AUXIO    26
C** GET I/O DEVICE - MUST BE INTEGER VARIABLE                             AUXIO    27
   40 CALL GNLE                                                           AUXIO    28
      IF(JTYP .NE. 2) GO TO 60                                            AUXIO    29
      IF(NEXT(JPTR) .NE. 1H ) GO TO 50                                    AUXIO    30
C** STORE IN SYMBOL TABLE                                                 AUXIO    31
      CALL SEARCH                                                         AUXIO    32
      IF(ISRCH(2) .EQ. 1) CALL ERROR(10,NXTID)                            AUXIO    33
      IF(ISRCH(1) .EQ. 1) GO TO 45                                        AUXIO    34
      IDTYP=1                                                             AUXIO    35
      CALL STORE                                                          AUXIO    36
      LOC=NID                                                             AUXIO    37
C** SET TYPE AND CHECK THAT IT IS INTEGER                                 AUXIO    38
   45 CALL IMPTYP                                                         AUXIO    39
      IF(BITGET(IDTBL(3,LOC),10,3) .NE. 4) CALL ERROR(22)                 AUXIO    40
      IF(BITGET(IDTBL(3,LOC),1,1) .EQ. 1) CALL ERROR(14,NXTID)            AUXIO    41
C** STORE IN BASIC BLOCK TABLE                                            AUXIO    42
      NBLOCK=NBLOCK+1                                                     AUXIO    43
      IBLOCK(NBLOCK)=2000+LOC                                             AUXIO    44
      RETURN                                                              AUXIO    45
   50 CALL ERROR(7)                                                       AUXIO    46
      RETURN                                                              AUXIO    47
   60 CALL ERROR(22)                                                      AUXIO    48
      RETURN                                                              AUXIO    49
      END                                                                 AUXIO    50
```

```
      INTEGER FUNCTION BITGET(ILOC,IPOS,IWIDTH)                          BITGET     2
      DIMENSION IMASK(18)                                                BITGET     3
      DATA (IMASK(I),I=1,18)/1B,3B,7B,17B,37B,77B,177B,377B,777B,        BITGET     4
     1 1777B,3777B,7777B,17777B,37777B,77777B,177777B,377777B,777777B,  BITGET     5
      BITGET=SHIFT(ILOC,IPOS) .AND. IMASK(IWIDTH)                        BITGET     6
      RETURN                                                             BITGET     7
      END                                                                BITGET     8
```

```
      INTEGER FUNCTION BITPUT(ILOC,IVAL,IPOS)                            BITPUT     2
      NSHIFT=60-IPOS                                                     BITPUT     3
      BITPUT=ILOC .OR. SHIFT(IVAL,NSHIFT)                                BITPUT     4
      RETURN                                                             BITPUT     5
      END                                                                BITPUT     6
```

```
      SUBROUTINE BLKSTR                                                 BLKSTR     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH       2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A     80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH       4
      COMMON/FUNC/IFNCRA(5,12),MARGS,IARGS(50),FNCLOC(5),NFUNC          CY58A     45
      COMMON/LIST/NLIST,NINTFC,ISUBLT(2,200),INTFAC(300)                BLKSTR     5
      COMMON/BASBLK/IBLOCK(2500),NBLOCK,NB,NBRNCH                       CY58A     46
      INTEGER BITPUT,BITGET,FNCLOC                                      BLKSTR     7
C** THIS ROUTINE IS CALLED AFTER PARSING AN EXPRESSION, TO STORE        BLKSTR     8
C** INFORMATION IN THE BASIC BLOCK TABLE                                BLKSTR     9
      IF(MARGS .EQ. 0) RETURN                                           BLKSTR    10
      DO 100 I=1,MARGS                                                  BLKSTR    11
      IOSTAT=2                                                          BLKSTR    12
      ICOL=20*MOD(I-1,3)+10                                             BLKSTR    13
      IVR=(I+2)/3                                                       BLKSTR    14
C** GET SYMBOL TABLE LOCATION OF NEXT VARIABLE IN THE STATEMENT         BLKSTR    15
      LOC=BITGET(IARGS(IVR),ICOL,10)                                    BLKSTR    16
      NFNC=BITGET(IARGS(IVR),ICOL+3,3)                                  BLKSTR    17
      IF(NFNC .EQ. 0) GO TO 60                                          BLKSTR    18
C** VARIABLE IS A FUNCTION ARGUMENT - GET SYMBOL TABLE LOCATION         BLKSTR    19
C** OF FUNCTION                                                        BLKSTR    20
      ILOC=FNCLOC(NFNC)                                                 BLKSTR    21
C** GET POSITION OF VARIABLE IN ARGUMENT LIST                           BLKSTR    22
      NARG=BITGET(IARGS(IVR),ICOL+9,6)                                  BLKSTR    23
C** GET SYMBOL TABLE LOCATION OF FUNCTION                               BLKSTR    24
      INDEX=BITGET(IDTBL(3,ILOC),36,9)                                  BLKSTR    25
      IF(INDEX .EQ. 0) GO TO 60                                         BLKSTR    26
C** GET INTERFACE DEFINITION TABLE POINTER                              BLKSTR    27
      IPTR=BITGET(ISUBLT(2,INDEX),60,15)+(NARG-1)/6                     BLKSTR    28
      JVAR=BITGET(ISUBLT(2,INDEX ),14,1)                                BLKSTR    29
      ICOL=9*MOD(NARG-1,6)+9                                            BLKSTR    30
      IF(JVAR .EQ. 1) ICOL=9                                            BLKSTR    31
C** GET I/O STATUS OF ARGUMENT                                          BLKSTR    32
      IOSTAT=BITGET(INTFAC(IPTR),ICOL,3)                                BLKSTR    33
      KPTR=(NARG+11)/6                                                  BLKSTR    34
      ICOL2=54+ICOL/9                                                   BLKSTR    35
C** SET "EXPRESSION" FLAG                                              BLKSTR    36
      IEXP=BITGET(IFNCRA(NFNC,KPTR),ICOL2,1)                            BLKSTR    37
      IF(IOSTAT .EQ. 2) GO TO 60                                        BLKSTR    38
      IF(IEXP .NE. 0) GO TO 40                                          BLKSTR    39
      IF(IOSTAT .EQ. 1) GO TO 60                                        BLKSTR    40
      GO TO 80                                                          BLKSTR    41
C** ARGUMENT APPEARS IN EXPRESSION BUT IS NOT DESIGNATED "LOGICAL       BLKSTR    42
C** INPUT" - MUST BE CLASS 0 FUNCTION                                   BLKSTR    43
C** IF CLASS 0 FUNCTION, CHANGE STATUS TO "LOGICAL INPUT" -             BLKSTR    44
C** OTHERWISE ISSUE DIAGNOSTIC                                          BLKSTR    45
   40 IF(BITGET(ISUBLT(2,INDEX),10,4) .NE. 0) GO TO 90                  BLKSTR    46
      INTFAC(IPTR)=BITPUT(INTFAC(IPTR),2,ICOL)                          BLKSTR    47
      IOSTAT=2                                                          BLKSTR    48
C** VARIABLE IS REFERENCED - STORE IN BASIC BLOCK TABLE                 BLKSTR    49
   60 NBLOCK=NBLOCK+1                                                   BLKSTR    50
      IBLOCK(NBLOCK)=2000+LOC                                           BLKSTR    51
      IF(IOSTAT .EQ. 2) GO TO 100                                       BLKSTR    52
C** VARIABLE IS DEFINED - STORE IN BASIC BLOCK TABLE                    BLKSTR    53
   80 NBLOCK=NBLOCK+1                                                   BLKSTR    54
      IBLOCK(NBLOCK)=1000+LOC                                           BLKSTR    55
      GO TO 100                                                         BLKSTR    56
   90 CALL ERROR(55,NARG)                                               BLKSTR    57
  100 CONTINUE                                                          BLKSTR    58
      RETURN                                                            BLKSTR    59
      END                                                               BLKSTR    60
```

```
      SUBROUTINE BUILD                                                  BUILD     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH      2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A    80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH      4
      COMMON/INPOUT/NCALL,IN,IOP                                        BUILD     4
      INTEGER A,B,BLANK                                                 BUILD     5
      COMMON/WASTE/B(72)                                                BUILD     6
C** THIS ROUTINE READS IN THE NEXT STATEMENT                            BUILD     7
      DATA BLANK/1H /,ICE/1HC/                                          BUILD     8
      IERR=0                                                            BUILD     9
      NFIRST=1                                                          BUILD    10
      NCONTU=0                                                          BUILD    11
      NCALL=NCALL+1                                                     BUILD    12
   50 CONTINUE                                                          BUILD    13
      IF(NFIRST .EQ. 1 .AND. NCALL .NE. 1) GO TO 1                      BUILD    14
C** READ NEXT CARD                                                      BUILD    15
      READ(IN,100) (B(I),I=1,72)                                        BUILD    16
      IF(EOF (IN) .NE. 0) GO TO 10                                      BUILD    17
  100 FORMAT(72A1)                                                      BUILD    18
    1 CONTINUE                                                          BUILD    19
      IF(NFIRST .EQ. 1) GO TO 2                                         BUILD    20
      IF(B(1) .EQ. ICE) GO TO 9                                         BUILD    21
      IF(B(6) .NE. BLANK .AND. B(6) .NE. 1H0) GO TO 6                   BUILD    22
      GO TO 9                                                           BUILD    23
    2 CONTINUE                                                          BUILD    24
C** STORE FIRST 72 COLUMNS                                              BUILD    25
      DO 3 I=1,72                                                       BUILD    26
      A(I)=B(I)                                                         BUILD    27
    3 CONTINUE                                                          BUILD    28
      NFIRST=0                                                          BUILD    29
      NCHAR=72                                                          BUILD    30
      GO TO 50                                                          BUILD    31
    6 NCONTU=NCONTU+1                                                   BUILD    32
      IF(NCONTU .LE. 19) GO TO 7                                        BUILD    33
      IERR=1                                                            BUILD    34
      CALL ERROR(4)                                                     BUILD    35
      RETURN                                                            BUILD    36
    7 CONTINUE                                                          BUILD    37
C** STORE COLUMNS 7-72 OF CONTINUATION CARD                             BUILD    38
      DO 8 I=1,66                                                       BUILD    39
    8 A(NCHAR+I)=B(I+6)                                                 BUILD    40
      NCHAR=NCHAR+66                                                    BUILD    41
      GO TO 50                                                          BUILD    42
   10 IERR=2                                                            BUILD    43
    9 CONTINUE                                                          BUILD    44
C** END OF STATEMENT, STORE NUMBER OF CHARACTERS                        BUILD    45
      N=NCHAR                                                           BUILD    46
      RETURN                                                            BUILD    47
      END                                                               BUILD    48
```

```
      SUBROUTINE CAA(ISTR,MSTR,ID)                                      CAA      2
      COMMON A(1326),D(500),IDTBL(6,500),INITID(3),LASTID(3),ISRCH(3),  RICH     2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IOTYP,NID,LOC,        CY58A    80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH     4
      DIMENSION ISTR(6)                                                 CAA      4
      DATA MASK/77000000000000000008/                                   CAA      5
      IF(MSTR .GT. 6 .AND. ITYP .NE. 28) CALL ERROR(6)                  CAA      6
      ID=0                                                              CAA      7
      NSHIFT=66                                                         CAA      8
      DO 20 I=1,10                                                      CAA      9
      NSHIFT=NSHIFT-6                                                   CAA     10
      IF(I .GT. MSTR) GO TO 10                                          CAA     11
      J=ISTR(I) .AND. MASK                                              CAA     12
      ID=ID .OR. SHIFT(J,NSHIFT)                                        CAA     13
      GO TO 20                                                          CAA     14
   10 ID=ID .OR. SHIFT(1L ,NSHIFT)                                      CAA     15
   20 CONTINUE                                                          CAA     16
      RETURN                                                            CAA     17
      END                                                               CAA     18
```

```
      SUBROUTINE CAI(ISTR,MSTR,INTVAL)                                  CAI      2
      DIMENSION ISTR(10)                                                CAI      3
      IF(MSTR .GT. 10) GO TO 20                                         CAI      4
      INTVAL=0                                                          CAI      5
      DO 10 I=1,MSTR                                                    CAI      6
      INT=(SHIFT(ISTR(I), 6) .AND. 77B) - 27                            CAI      7
      IF(INT .EQ. 0) GO TO 10                                           CAI      8
      INTVAL=INTVAL+INT*10**(MSTR-I)                                    CAI      9
   10 CONTINUE                                                          CAI     10
      IF(INTVAL .GT. (2**31-1)) GO TO 20                                CAI     11
      RETURN                                                            CAI     12
   20 CALL ERROR(3)                                                     CAI     13
      RETURN                                                            CAI     14
      END                                                               CAI     15
```

```
      SUBROUTINE CALL                                                    CALL      2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),   RICH      2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,         CY58A    80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                   RICH      4
      COMMON/STRING/NTYPE,NSTR,STR(500)                                  CALL      4
      COMMON/LIST/NLIST,NINTFC,ISUBLT(2,200),INTFAC(300)                 CALL      5
      DIMENSION IALPH(4)                                                 CALL      6
      INTEGER BLANK,BITPUT,BITGET                                        CALL      7
      DATA (IALPH(I),I=1,4)/1HC,1HA,1HL,1HL/                             CALL      8
      DATA LPAR/1H(/,BLANK/1H /                                          CALL      9
C** CALL STATEMENT PROCESSOR                                             CALL     10
      DO 5 I=1,4                                                         CALL     11
      IF(NEXT(JPTR) .NE. IALPH(I)) GO TO 50                              CALL     12
    5 CONTINUE                                                           CALL     13
      IPTR=JPTR                                                          CALL     14
C** GET SUBROUTINE NAME                                                  CALL     15
      CALL GNLE                                                          CALL     16
      IF(JTYP .NE. 2) GO TO 50                                           CALL     17
      IF(NXTID .EQ. IFNCNM) CALL ERROR(10,NXTID)                         CALL     18
C** STORE IN SYMBOL TABLE                                                CALL     19
      CALL SEARCH                                                        CALL     20
      IF(ISRCH(1) .EQ. 1) CALL ERROR(24,NXTID)                           CALL     21
      IF(ISRCH(2) .EQ. 1) GO TO 8                                        CALL     22
      IDTYP=2                                                            CALL     23
      CALL STORE                                                         CALL     24
      LOC=NID                                                            CALL     25
    8 CONTINUE                                                           CALL     26
      ILOC=LOC                                                           CALL     27
      NXT=NEXT(JPTR)                                                     CALL     28
      IF(NXT .EQ. LPAR) GO TO 17                                         CALL     29
      IF(NXT .NE. BLANK) GO TO 50                                        CALL     30
C** SUBROUTINE HAS NO ARGUMENT LIST                                      CALL     31
C** CHECK IF NAME HAS APPEARED PREVIOUSLY                                CALL     32
      IF(BITGET(IDTBL(3,LOC),18,1) .EQ. 1) GO TO 20                      CALL     33
C** NAME HAS NOT YET APPEARED                                            CALL     34
C** SET "APPEARED" FLAG                                                  CALL     35
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),1,18)                             CALL     36
C** SEARCH THE SESCOMP LIST                                              CALL     37
      DO 10 I=1,NLIST                                                    CALL     38
      IF(IDTBL(1,LOC) .NE. ISUBLT(1,I)) GO TO 10                         CALL     39
C** NAME FOUND IN LIST                                                   CALL     40
C** STORE LIST LOCATION IN SYMBOL TABLE                                  CALL     41
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),I,36)                             CALL     42
      LISTLC=I                                                           CALL     43
      GO TO 22                                                           CALL     44
   10 CONTINUE                                                           CALL     45
C** NAME IS NOT IN SESCOMP LIST                                          CALL     46
C** PUT NAME IN LIST AND ISSUE WARNING MESSAGE                           CALL     47
      NLIST=NLIST+1                                                      CALL     48
      ISUBLT(1,NLIST)=IDTBL(1,LOC)                                       CALL     49
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),NLIST,36)                         CALL     50
      CALL ERROR(52)                                                     CALL     51
      RETURN                                                             CALL     52
C** NAME HAS PREVIOUSLY APPEARED - GET LIST LOCATION                     CALL     53
   20 LISTLC=BITGET(IDTBL(3,LOC),36,9)                                   CALL     54
   22 CONTINUE                                                           CALL     55
C** CHECK THAT NUMBER OF ARGUMENTS IS ZERO                               CALL     56
```

```
      IF(BITGET(ISUBLT(2,LISTLC),6,6) .NE. 0) CALL ERROR(26)
      RETURN
C** SUBROUTINE HAS AN ARGUMENT LIST
   17 JPTR=IPTR
      NTYPE=1
C** PARSE THE STATEMENT
      CALL EXPR
      CALL PARSE
C** PROCESS ALL EXTERNAL REFERENCES
      CALL FNCSTR
C** STORE BASIC BLOCKS
      CALL BLKSTR
      IF(MODE .EQ. 1) GO TO 48
C** ROLL CALL MODE - MAY HAVE TO ISSUE A CALL TO "ROLCHK"
      LOC=ILOC
C** GET SUBROUTINE CLASS
      INDEX=BITGET(IDTBL(3,LOC),36,9)
      KLAS=BITGET(ISUBLT(2,INDEX),10,4)
C** IF SESCOMP MODULE - ISSUE A CALL TO "ROLCHK"
      IF(KLAS .EQ. 1 .OR. KLAS .EQ. 2) CALL CALL2
      RETURN
C** VARIABLE PRECISION MODE
C** ISSUE CALLS TO VARIABLE PRECISION SUBROUTINES
   48 JPTR=IPTR-1
      CALL CNVRT
      RETURN
   50 CALL ERROR(7)
      RETURN
      END
```

```
      SUBROUTINE CALL2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES
      INTEGER A,D,COMMA,RPAR
      INTEGER BITPUT,BITGET
      DIMENSION IALPH(13)
      DATA RPAR/1H)/,COMMA/1H,/
      DATA (IALPH(I),I=1,13)/1HC,1HA,1HL,1HL,1H ,1HR,1HO,1HL,1HC,1HH,
     1 1HK,1H ,1H(/
C** THIS ROUTINE GENERATES A CALL TO "ROLCHK" OF THE FORM
C**       CALL ROLCHK(1HN,1HA,1HM,1HE,1H ,1H )
      DO 15 J=1,13
      K=J+6
      A(K)=IALPH(J)
   15 CONTINUE
C** GENERATE ARGUMENT LIST
      DO 20 I=1,6
      KK=19+4*I
      A(KK-3)=1H1
      A(KK-2)=1HH
      IPOS=6*I
      IVL=BITGET(IDTBL(1,LOC),IPOS,6)
      A(KK-1)=BITPUT(0,IVL,6)
      IF(I .EQ. 6) GO TO 25
   20 A(KK)=COMMA
   25 A(KK)=RPAR
      N=KK
      RETURN
      END
```

```
      SUBROUTINE CAR(RN,NCHAR,IDES)                                     CAR         2
      DIMENSION IHOL(4)                                                 CAR         3
      INTEGER RN(40),BLANK,CHAR                                         CAR         4
      DOUBLE PRECISION DPVAL                                            CAR         5
      DATA RMAX,RMIN,BLANK/1.0E+38,1.0E-38,5500000000000000000008/      CAR         6
      IF(NCHAR .GT. 20*(IDES+1)) RETURN                                 CAR         7
      KOUNT=0                                                           CAR         8
      ICHAR=NCHAR+1                                                     CAR         9
      DO 5 I=1,4                                                        CAR        10
    5 IHOL(I)=0                                                         CAR        11
      NW=2*IDES+3                                                       CAR        12
   10 NW=NW-1                                                           CAR        13
      NSHIFT=0                                                          CAR        14
   15 KOUNT=KOUNT+1                                                     CAR        15
      IF(KOUNT .GT. NCHAR) GO TO 20                                     CAR        16
      ICHAR=ICHAR-1                                                     CAR        17
      CHAR=RN(ICHAR) .AND. 77000000000000000008                         CAR        18
      GO TO 25                                                          CAR        19
   20 CHAR=BLANK                                                        CAR        20
   25 NSHIFT=NSHIFT+6                                                   CAR        21
      IHOL(NW)=IHOL(NW) .OR. SHIFT(CHAR,NSHIFT)                         CAR        22
      IF(NSHIFT .LT. 60) GO TO 15                                       CAR        23
      IF(NW .GT. 1) GO TO 10                                            CAR        24
      IF(IDES .EQ. 1) GO TO 35                                          CAR        25
      DECODE(20,30,IHOL(1)) RVAL                                        CAR        26
   30 FORMAT(E20.0)                                                     CAR        27
      IF(RVAL .EQ. 0.0) RETURN                                          CAR        28
      IF(RVAL .GT. RMAX .OR. RVAL .LT. RMIN) CALL ERROR(46)             CAR        29
      RETURN                                                            CAR        30
   35 DECODE(40,40,IHOL(1)) DPVAL                                       CAR        31
   40 FORMAT(D40.0)                                                     CAR        32
      IF(DPVAL .EQ. 0.0) RETURN                                         CAR        33
      IF(DPVAL .GT. RMAX .OR. DPVAL .LT. RMIN) CALL ERROR(46)           CAR        34
      RETURN                                                            CAR        35
      END                                                               CAR        36
```

```
      SUBROUTINE CHKLST                                                 CHKLIST    2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH       2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A     80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH       4
      DIMENSION IQUIV(100)                                              CHKLIST    4
      EQUIVALENCE(IQUIV(1),A(301))                                      CHKLIST    5
      INTEGER BITGET                                                    CHKLIST    6
C** THIS ROUTINE IS CALLED PRIOR TO THE FLOW ANALYSIS TO FLAG           CHKLIST    7
C** ALL INITIALLY DEFINED VARIABLES                                     CHKLIST    8
      NQUIV=0                                                           CHKLIST    9
      DO 30 I=1,NID                                                     CHKLIST   10
      IDTBL(8,I)=0                                                      CHKLIST   11
      IF(BITGET(IDTBL(3,I),14,1) .EQ. 1) GO TO 20                       CHKLIST   12
      IF(BITGET(IDTBL(3,I),16,1) .EQ. 1) GO TO 5                        CHKLIST   13
      IF(BITGET(IDTBL(3,I),12,1) .EQ. 1) GO TO 15                       CHKLIST   14
      GO TO 10                                                          CHKLIST   15
C** VARIABLE IN COMMON - SET INITIALLY DEFINED FLAG                     CHKLIST   16
    5 IDTBL(8,I)=1                                                      CHKLIST   17
   10 IF(BITGET(IDTBL(3,I),17,1) .NE. 1) GO TO 30                       CHKLIST   18
C** VARIABLE IS EQUIVALENCED - STORE IN LIST                            CHKLIST   19
      NQUIV=NQUIV+1                                                     CHKLIST   20
      IF(NQUIV .GT. 100) GO TO 60                                       CY58A     52
      IQUIV(NQUIV)=I                                                    CHKLIST   21
      GO TO 30                                                          CHKLIST   22
C** VARIABLE IS FORMAL PARAMETER - FLAG IF INPUT                        CHKLIST   23
   15 IF(BITGET(IDTBL(3,I),37,1) .EQ. 0) GO TO 30                       CHKLIST   24
C** VARIABLE IS DEFINED BY DATA STATEMENT OR IS INPUT                   CHKLIST   25
   20 IDTBL(8,I)=1                                                      CHKLIST   26
   30 CONTINUE                                                          CHKLIST   27
      IF(NQUIV .EQ. 0) RETURN                                           CHKLIST   28
      DO 50 J=1,NQUIV                                                   CHKLIST   29
      NXQV=IQUIV(J)                                                     CHKLIST   30
C** GET NEXT EQUIVALENCED VARIABLE                                      CHKLIST   31
   35 NXQV=IDTBL(7,NXQV)                                                CHKLIST   32
      IF(NXQV .EQ. IQUIV(J)) GO TO 50                                   CHKLIST   33
      IF(IDTBL(8,NXQV) .EQ. 0) GO TO 35                                 CHKLIST   34
C** VARIABLE IN EQUIVALENCE LINK IS DEFINED                             CHKLIST   35
      IQV=NXQV                                                          CHKLIST   36
C** CHECK TYPE OF DEFINED VARIABLE                                      CHKLIST   37
      KTYPE=BITGET(IDTBL(3,IQV),10,3)                                   CHKLIST   38
   40 NXQV=IDTBL(7,NXQV)                                                CHKLIST   39
      IF(NXQV .EQ. IQV) GO TO 50                                        CHKLIST   40
      IF(BITGET(IDTBL(3,NXQV),10,3) .NE. KTYPE) GO TO 40                CHKLIST   41
C** VARIABLE IN EQUIVALENCE LINK HAS SAME TYPE - SET FLAG TO "DEFINED"  CHKLIST   42
      IDTBL(8,NXQV)=1                                                   CHKLIST   43
      GO TO 40                                                          CHKLIST   44
   50 CONTINUE                                                          CHKLIST   45
      RETURN                                                            CHKLIST   46
   60 CALL ERROR(94)                                                    CY58A     53
      RETURN                                                            CY58A     54
      END                                                               CHKLIST   47
```

```
      SUBROUTINE CLASS                                                  CLASS      2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH       2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A     80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH       4
      DIMENSION KALP(48),KSUC(48),KFAL(48),KDEC(10),KF(8)               CLASS      4
      INTEGER A                                                         CLASS      5
C** THIS ROUTINE CLASSIFIES FORTRAN STATEMENTS INTO 36 CLASSES          CLASS      6
C** AND STORES THE CLASS IN "ITYP"                                      CLASS      7
C**         1-ASSIGNMENT   2-ASSIGN       3-GO TO         4-ASSD. GO TO CLASS      8
C**         5-COMP. GO TO  6-ARITH. IF    7-CONTINUE      8-CALL        CLASS      9
C**         9-RETURN       10-STOP        11-READ         12-WRITE      CLASS     10
C**         13-REWIND      14-BACKSPACE   15-ENDFILE      16-LOGICAL IF CLASS     11
C**         17-DO          18-END         19-INTEGER      20-REAL       CLASS     12
C**         21-DOUB. PREC. 22-COMPLEX     23-LOGICAL      24-DIMENSION  CLASS     13
C**         25-COMMON      26-EQUIVALENCE 27-DATA         28-FORMAT     CLASS     14
C**         29-BLOCK DATA  30-SUBROUTINE  31-FUNCTION     32-PROGRAM    CLASS     15
C**         33-36  INVALID                                              CLASS     16
      DATA KDEC(1),KDEC(2),KDEC(3),KDEC(4),KDEC(5),                     CLASS     17
     1  KDEC(6),KDEC(7),KDEC(8),KDEC(9),KDEC(10)                        CLASS     18
     2  /1H0,1H1,1H2,1H3,1H4,1H5,1H6,1H7,1H8,1H9/                       CLASS     19
      DATA KF(1),KF(2),KF(3),KF(4),KF(5),KF(6),KF(7),KF(8)              CLASS     20
     1   /1HF,1HU,1HN,1HC,1HT,1HI,1HO,1HN/                            CLASS     21
      DATA KC,KBLNK,KLPAR,KRPAR,KEQ  /1HC,1H ,1H(,1H),1H=/              CLASS     22
      DATA KH,KSLSH,KASTK,KZERO,KCMA  /1HH,1H/,1H*,1H0,1H,/             CLASS     23
C** CHARACTER ARRAY FOR TREE SCAN                                       CLASS     24
      DATA KALP( 1),KALP( 2),KALP( 3),KALP( 4) /1HI,1HF,1HN,1HG/        CLASS     25
      DATA KALP( 5),KALP( 6),KALP( 7),KALP( 8) /1HO,1HT,1HO,1H(/        CLASS     26
      DATA KALP( 9),KALP(10),KALP(11),KALP(12) /1HC,1HA,1HO,1HN/        CLASS     27
      DATA KALP(13),KALP(14),KALP(15),KALP(16) /1HH,1HH,1HP,1HR/        CLASS     28
      DATA KALP(17),KALP(18),KALP(19),KALP(20) /1HE,1HA,1HD,1HL/        CLASS     29
      DATA KALP(21),KALP(22),KALP(23),KALP(24) /1HT,1HW,1HF,1HO/        CLASS     30
      DATA KALP(25),KALP(26),KALP(27),KALP(28) /1HU,1HD,1HI,1HA/        CLASS     31
      DATA KALP(29),KALP(30),KALP(31),KALP(32) /1HO,1HU,1HW,1HS/        CLASS     32
      DATA KALP(33),KALP(34),KALP(35),KALP(36) /1HT,1HU,1HE,1HN/        CLASS     33
      DATA KALP(37),KALP(38),KALP(39),KALP(40) /1HD,1HF,1HX,1HO/        CLASS     34
      DATA KALP(41),KALP(42),KALP(43),KALP(44) /1HB,1HA,1HL,1HA/        CLASS     35
      DATA KALP(45),KALP(46),KALP(47),KALP(48) /1HL,1HP,1HR,1HO/        CLASS     36
C** SUCCEED LINK FOR TREE SCAN                                          CLASS     37
      DATA KSUC( 1),KSUC( 2),KSUC( 3),KSUC( 4) /  2, -6,-19,  5/       CLASS     38
      DATA KSUC( 5),KSUC( 6),KSUC( 7),KSUC( 8) /  6,  7,  8, -5/       CLASS     39
      DATA KSUC( 9),KSUC(10),KSUC(11),KSUC(12) / 10, -8, 12, -7/       CLASS     40
      DATA KSUC(13),KSUC(14),KSUC(15),KSUC(16) / 14,-25,-22, 17/       CLASS     41
      DATA KSUC(17),KSUC(18),KSUC(19),KSUC(20) / 18, 19,-11,-20/       CLASS     42
      DATA KSUC(21),KSUC(22),KSUC(23),KSUC(24) / -9,-13, 24,-28/       CLASS     43
      DATA KSUC(25),KSUC(26),KSUC(27),KSUC(28) /-31, 27,-24,-27/       CLASS     44
      DATA KSUC(29),KSUC(30),KSUC(31),KSUC(32) / 30,-21,-12, 33/       CLASS     45
      DATA KSUC(33),KSUC(34),KSUC(35),KSUC(36) /-10,-30, 36, 37/       CLASS     46
      DATA KSUC(37),KSUC(38),KSUC(39),KSUC(40) / 38,-15,-34,-26/       CLASS     47
      DATA KSUC(41),KSUC(42),KSUC(43),KSUC(44) / 42,-14,-29, -2/       CLASS     48
      DATA KSUC(45),KSUC(46),KSUC(47),KSUC(48) /-23,47,48,-32/         CLASS     49
C** FAIL LINK FOR TREE SCAN                                             CLASS     50
      DATA KFAL( 1),KFAL( 2),KFAL( 3),KFAL( 4) /  4,  3,-36,  9/       CLASS     51
      DATA KFAL( 5),KFAL( 6),KFAL( 7),KFAL( 8) /-36,-36,-36, -3/       CLASS     52
      DATA KFAL( 9),KFAL(10),KFAL(11),KFAL(12) / 16, 11,-36, 13/       CLASS     53
      DATA KFAL(13),KFAL(14),KFAL(15),KFAL(16) /-36, 15,-36, 23/       CLASS     54
      DATA KFAL(17),KFAL(18),KFAL(19),KFAL(20) /-36, 21, 20,-36/       CLASS     55
      DATA KFAL(21),KFAL(22),KFAL(23),KFAL(24) / 22,-36, 26, 25/       CLASS     56
```

```
      DATA KFAL(25),KFAL(26),KFAL(27),KFAL(28) /-36, 31, 28, 29/          CLASS      57
      DATA KFAL(29),KFAL(30),KFAL(31),KFAL(32) /-36,-36, 32, 35/          CLASS      58
      DATA KFAL(33),KFAL(34),KFAL(35),KFAL(36) / 34,-36, 41, 39/          CLASS      59
      DATA KFAL(37),KFAL(38),KFAL(39),KFAL(40) /-36,-18, 40,-36/          CLASS      60
      DATA KFAL(41),KFAL(42),KFAL(43),KFAL(44) / 44, 43,-36, 45/          CLASS      61
      DATA KFAL(45),KFAL(46),KFAL(47),KFAL(48) / 46,-36,-36,-36/          CLASS      62
      LTYP=0                                                              CLASS      63
      IPTR=7                                                              CLASS      64
    5 CONTINUE                                                            CLASS      65
      JSAVE=KBLNK                                                         CLASS      66
      JSW=0                                                               CLASS      67
      ISW=0                                                               CLASS      68
      JEQ=0                                                               CLASS      69
      JCMA=0                                                              CLASS      70
      JHOLL=0                                                             CLASS      71
C** ASSIGNMENT SCAN LOOP                                                  CLASS      72
      DO 26 J=IPTR,N                                                      CLASS      73
      JCH=A(J)                                                            CLASS      74
      IF(JCH .EQ. KBLNK) GO TO 26                                         CLASS      75
C** IF NOT BLANK, CHECK FOR HOLLERITH SWITCH                              CLASS      76
      IF(JHOLL .LE. 0) GO TO 12                                           CLASS      77
      DO 8 L=1,10                                                         CLASS      78
      IF(JCH .EQ. KDEC(L)) GO TO 10                                       CLASS      79
    8 CONTINUE                                                            CLASS      80
C** FIRST TIME, NO INTEGER MEANS NOT HOLLERITH                            CLASS      81
      IF(JHOLL .LE. 1) GO TO 11                                           CLASS      82
C** OTHERWISE LOOK FOR "H"                                                CLASS      83
      IF(JCH-KH) 11,32,11                                                 CLASS      84
C** STILL FITS HOLLERITH SYNTAX                                           CLASS      85
   10 JHOLL=JHOLL+1                                                       CLASS      86
      GO TO 25                                                            CLASS      87
C** NOT A HOLLERITH CONSTANT, SET SWITCH OFF                              CLASS      88
   11 JHOLL=0                                                             CLASS      89
C** TEST OTHER CHARACTERS (),=/*                                          CLASS      90
   12 IF(JCH .EQ. KLPAR) GO TO 20                                         CLASS      91
      IF(JCH .EQ. KRPAR) GO TO 18                                         CLASS      92
      IF(JCH .EQ. KCMA) GO TO 22                                          CLASS      93
      IF(JCH .EQ. KEQ) GO TO 23                                           CLASS      94
      IF(JCH .EQ. KSLSH) GO TO 21                                         CLASS      95
      IF(JCH-KASTK) 25,21,25                                              CLASS      96
C** RIGHT PAREN FOUND                                                     CLASS      97
   18 JSW=JSW-1                                                           CLASS      98
      IF(JSW .GT. 0) GO TO 25                                             CLASS      99
C** SET SWITCH TO ALLOW ONLY ONE MORE NON-BLANK CHARACTER                  CLASS     100
      ISW=1                                                               CLASS     101
      GO TO 26                                                            CLASS     102
C** LEFT PAREN FOUND                                                      CLASS     103
   20 JSW=JSW+1                                                           CLASS     104
C** SET HOLLERITH SWITCH FOR (,/*                                         CLASS     105
   21 JHOLL=1                                                             CLASS     106
      GO TO 25                                                            CLASS     107
C** COMMA FOUND, CHECK LEVEL                                              CLASS     108
   22 IF(JSW) 30,30,21                                                    CLASS     109
C** EQUAL SIGN FOUND, CHECK LEVEL                                         CLASS     110
   23 IF(JSW .GT. 0) GO TO 32                                             CLASS     111
      JEQ=1                                                               CLASS     112
C** TEST IF TERMINATED BY SWITCH SET                                      CLASS     113
```

```
   25 IF(ISW .GT. 0) GO TO 27                                    CLASS     114
   26 CONTINUE                                                   CLASS     115
      GO TO 28                                                   CLASS     116
C** SAVE LAST CHARACTER IF TERMINATED EARLY                      CLASS     117
   27 JSAVE=JCH                                                  CLASS     118
C** ONE NON-BLANK CHARACTER AFTER A RIGHT PAREN                  CLASS     119
C** MIGHT BE AN ASSIGNMENT                                       CLASS     120
      JP=J                                                       CLASS     121
   28 IF(JEQ .LE. 0) GO TO 32                                    CLASS     122
      JTP=1                                                      CLASS     123
      GO TO 55                                                   CLASS     124
C** UPPER LEVEL COMMA FOUND                                      CLASS     125
C** MIGHT BE A DO                                                CLASS     126
   30 JCMA=1                                                     CLASS     127
      IF(JEQ .LE. 0) GO TO 32                                    CLASS     128
      JTP=17                                                     CLASS     129
      GO TO 55                                                   CLASS     130
C** HOLLERITH CONSTANT FOUND                                     CLASS     131
   32 J=1                                                        CLASS     132
      ISW=IPTR                                                   CLASS     133
   33 JCH=A(ISW)                                                 CLASS     134
      IF(JCH .EQ. KBLNK) GO TO 37                                CLASS     135
C** TEST AGAINST CURRENT TREE CHARACTER                          CLASS     136
   34 IF(JCH .EQ. KALP(J)) GO TO 36                              CLASS     137
C** IF NO MATCH, TRY NEXT IN TREE                                CLASS     138
   35 J=KFAL(J)                                                  CLASS     139
      IF (J) 39,39,34                                            CLASS     140
C** CHARACTER MATCHES, TRY NEXT IN TREE                          CLASS     141
   36 J=KSUC(J)                                                  CLASS     142
      IF(J .LE. 0) GO TO 39                                      CLASS     143
   37 ISW=ISW+1                                                  CLASS     144
      IF(ISW .LE. N) GO TO 33                                    CLASS     145
C** RUN OUT OF CHARACTERS                                        CLASS     146
      JCH=KBLNK                                                  CLASS     147
      GO TO 35                                                   CLASS     148
C** CLASSIFICATION COMPLETED, FORM TYPE CODE                     CLASS     149
   39 JTP=-J                                                     CLASS     150
C** CHECK TO SEE IF MORE TREATMENT NEEDED                        CLASS     151
      IF(JTP-3) 55,45,40                                         CLASS     152
   40 IF(JTP-6) 55,43,41                                         CLASS     153
   41 IF(JTP .LT. 19) GO TO 55                                   CLASS     154
      IF(JTP-23) 47,47,55                                        CLASS     155
C** LOGICAL IF SEPARATION TEST                                   CLASS     156
   43 DO 44 L=1,10                                               CLASS     157
      IF(JSAVE .EQ. KDEC(L)) GO TO 55                            CLASS     158
   44 CONTINUE                                                   CLASS     159
      LTYP=9                                                     CLASS     160
      JTP=16                                                     CLASS     161
      IPTR=JP                                                    CLASS     162
      GO TO 5                                                    CLASS     163
C** SEPARATE ASSIGNED AND UNCONDITIONAL GOTOS                    CLASS     164
   45 IF(JCMA .LE. 0) GO TO 55                                   CLASS     165
      JTP=4                                                      CLASS     166
      GO TO 55                                                   CLASS     167
C** CHECK WHETHER THIS IS A TYPE STATEMENT OR TYPED FUNCTION     CLASS     168
   47 L=11                                                       CLASS     169
      GO TO 52                                                   CLASS     170
   48 L=L+1                                                      CLASS     171
      IF(L .GT. N) GO TO 55                                      CLASS     172
      IF(A(L) .EQ. KBLNK) GO TO 48                               CLASS     173
   50 IF(A(L) .EQ. KF(ISW)) GO TO 53                             CLASS     174
      IF(ISW .EQ. 1) GO TO 48                                    CLASS     175
   52 ISW=1                                                      CLASS     176
      GO TO 50                                                   CLASS     177
   53 ISW=ISW+1                                                  CLASS     178
      IF(ISW .LE. 8) GO TO 48                                    CLASS     179
      JTP=31                                                     CLASS     180
   55 ITYP=JTP                                                   CLASS     181
C** ALL RESULTS COME HERE FOR RETURN                             CLASS     182
      RETURN                                                     CLASS     183
      END                                                        CLASS     184
```

```
      SUBROUTINE CMPARE
      DIMENSION IROLCK(100),ISUB(100)
      REWIND 3
      NSUB=0
      NC=0
      DO 5 I=1,100
      READ(9) ISUB(I)
      IF(EOF(9) .NE. 0) GO TO 7
      NSUB=NSUB+1
    5 CONTINUE
    7 IF(NSUB .EQ. 0) RETURN
      DO 40 L=1,12
      MODE=L-1
      NROLL=0
      DO 10 I=1,100
      READ(3) IROLCK(I)
      IF(EOF(3) .NE. 0) GO TO 15
      NROLL=NROLL+1
   10 CONTINUE
   15 IF(NROLL .EQ. 0) GO TO 35
      DO 30 J=1,NSUB
      DO 20 K=1,NROLL
      IF(IROLCK(K) .EQ. ISUB(J)) GO TO 30
   20 CONTINUE
      NC=1
      WRITE(6,25) ISUB(J),MODE
   25 FORMAT(/29X,12H SUBROUTINE ,A6,53H WAS NOT CALLED IN THE ROLL CALL
     * MODE FOR MODE INDEX ,I3)
   30 CONTINUE
      GO TO 40
   35 WRITE(6,36) MODE
   36 FORMAT(/32X,65H NO SUBROUTINES WERE CALLED IN THE ROLL CALL MODE F
     *OR MODE INDEX ,I3)
      NC=1
   40 CONTINUE
      IF(NC .EQ. 0) WRITE(6,50)
   50 FORMAT(/41X,50H ALL SUBROUTINES WERE CALLED IN THE ROLL CALL MODE)
      RETURN
      END
```

```
      SUBROUTINE CNVRT                                                  CNVRT      2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH       2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A     80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDFS                                  RICH       4
      COMMON/STRING/NTYPE,NSTR,STR(500)                                 CNVRT      4
      DIMENSION IOP(8),ILOG(7,2),JLOG(2,3),IFUNC1(6),IFUNC2(6),         CNVRT      5
     1 IFUNC3(6)                                                        CNVRT      6
      INTEGER STR,A,D,DECPT                                             CNVRT      7
      INTEGER BITPUT,BITGET                                             CNVRT      8
      DATA (IOP(I),I=1,8)/1H+,1H-,1H/,1H(,1H),1H,,1H*,1H*/              CNVRT      9
      DATA (((ILOG(I,J),J=1,2),I=1,7)/1HL,1HT,1HL,1HE,1HG,1HT,1HG,1HE,  CNVRT     10
     1 1HE,1HQ,1HN,1HE,1HO,1HR/                                         CNVRT     11
      DATA ((JLOG(I,J),J=1,3),I=1,2)/1HA,1HN,1HD,1HN,1HO,1HT/           CNVRT     12
      DATA (IFUNC1(I),I=1,6)/1HQ,1H1,1HR,1HE,1HA,1HL/                   CNVRT     13
      DATA (IFUNC2(I),I=1,6)/1HQ,1H1,1HD,1HP,1HR,1HE/                   CNVRT     14
      DATA (IFUNC3(I),I=1,6)/1HQ,1H1,1HC,1HO,1HM,1HP/                   CNVRT     15
      DATA DECPT/1H./                                                   CNVRT     16
C** THIS ROUTINE IS CALLED IN THE VARIABLE PRECISION MODE TO DECODE     CNVRT     17
C** THE STRING WHICH IS RETURNED BY THE PARSER AFTER VARIABLE           CNVRT     18
C** PRECISION CALLS HAVE BEEN INSERTED                                 CNVRT     19
      DO 5 J=1,JPTR                                                     CNVRT     20
    5 D(J)=A(J)                                                         CNVRT     21
C** THIS LOOP LOOKS AT EVERY STRING ELEMENT INDIVIDUALLY                CNVRT     22
      J=JPTR                                                            CNVRT     23
      DO 100 K=1,NSTR                                                   CNVRT     24
      IF(STR(K) .GT. 0) GO TO 40                                        CNVRT     25
C** SPECIAL CHARACTER OR OPERATOR                                       CNVRT     26
      DO 10 I=1,8                                                       CNVRT     27
      IF(STR(K) .NE. -I) GO TO 10                                       CNVRT     28
C** LOGICAL OPERATOR FOUND - STORE IN "D"                               CNVRT     29
C** CHARACTER FOUND - STORE IN "D"                                      CNVRT     30
      D(J+1)=IOP(I)                                                     CNVRT     31
      N3=1                                                              CNVRT     32
      IF(I .NE. 8) GO TO 100                                            CNVRT     33
      D(J+2)=IOP(I)                                                     CNVRT     34
      N3=2                                                              CNVRT     35
      GO TO 100                                                         CNVRT     36
   10 CONTINUE                                                          CNVRT     37
      DO 15 I=1,7                                                       CNVRT     38
      L=I+8                                                             CNVRT     39
      IF(STR(K) .NE. -L) GO TO 15                                       CNVRT     40
C** LOGICAL OPERATOR FOUND - STORE IN "D"                               CNVRT     41
      D(J+1)=DECPT                                                      CNVRT     42
      D(J+2)=ILOG(I,1)                                                  CNVRT     43
      D(J+3)=ILOG(I,2)                                                  CNVRT     44
      D(J+4)=DECPT                                                      CNVRT     45
      N3=4                                                              CNVRT     46
      GO TO 100                                                         CNVRT     47
   15 CONTINUE                                                          CNVRT     48
      DO 20 I=1,2                                                       CNVRT     49
      L=I+15                                                            CNVRT     50
      IF(STR(K) .NE. -L) GO TO 20                                       CNVRT     51
      D(J+1)=DECPT                                                      CNVRT     52
      D(J+2)=JLOG(I,1)                                                  CNVRT     53
      D(J+3)=JLOG(I,2)                                                  CNVRT     54
      D(J+4)=JLOG(I,3)                                                  CNVRT     55
      D(J+5)=DECPT                                                      CNVRT     56
```

```
      N3=5                                                              CNVRT      57
      GO TO 100                                                         CNVRT      58
   20 CONTINUE                                                          CNVRT      59
C** MUST BE A CALL TO A VARIABLE PRECISION SUBROUTINE                   CNVRT      60
      KL=1                                                              CNVRT      61
      IF(STR(K) .EQ. -0) KL=2                                           CNVRT      62
      IF(STR(K) .EQ. -10000) KL=4                                       CNVRT      63
      IF(STR(K) .EQ. -20000) KL=3                                       CNVRT      64
      GO TO(110,25,30,35),KL                                            CNVRT      65
C** CALL TO "Q1REAL" - STORE IN "D"                                     CNVRT      66
   25 DO 27 I=1,6                                                       CNVRT      67
      D(J+I)=IFUNC1(I)                                                  CNVRT      68
   27 CONTINUE                                                          CNVRT      69
      N3=6                                                              CNVRT      70
      GO TO 100                                                         CNVRT      71
C** CALL TO "Q1DPRE" - STORE IN "D"                                     CNVRT      72
   30 DO 32 I=1,6                                                       CNVRT      73
      D(J+I)=IFUNC2(I)                                                  CNVRT      74
   32 CONTINUE                                                          CNVRT      75
      N3=6                                                              CNVRT      76
      GO TO 100                                                         CNVRT      77
C** CALL TO "Q1COMP" - STORE IN "D"                                     CNVRT      78
   35 DO 37 I=1,6                                                       CNVRT      79
      D(J+I)=IFUNC3(I)                                                  CNVRT      80
   37 CONTINUE                                                          CNVRT      81
      N3=6                                                              CNVRT      82
      GO TO 100                                                         CNVRT      83
   40 IF(STR(K) .LT. 1000001) GO TO 110                                 CNVRT      84
      N3=STR(K)/1.E6                                                    CNVRT      85
      NLOC=(STR(K)/10000)*10000                                         CNVRT      86
      JPTR=STR(K)-NLOC                                                  CNVRT      87
      KLOC=STR(K)-N3*1.E6                                               CNVRT      88
      IF(KLOC .LT. 400000 .OR. KLOC .GT. 500000) GO TO 50               CNVRT      89
      IHL=(KLOC-400000)/10000                                           CY61        1
      IF(IHL .EQ. 5) GO TO 60                                           CY61        2
C** CONSTANT FOUND - STORE IN "D"                                       CNVRT      90
      DO 45 I=1,N3                                                      CNVRT      91
      D(J+I)=NEXT(JPTR)                                                 CNVRT      92
   45 CONTINUE                                                          CNVRT      93
      GO TO 100                                                         CNVRT      94
C** VARIABLE FOUND - STORE IN "D"                                       CNVRT      95
   50 DO 55 I=1,N3                                                      CNVRT      96
      IPOS=6*I                                                          CNVRT      97
      ICHAR=BITGET(IDTBL(1,JPTR),IPOS,6)                                CNVRT      98
   55 D(J+I)=BITPUT(0,ICHAR,6)                                          CNVRT      99
      GO TO 100                                                         CY61        3
   60 KPTR=JPTR-1                                                       CY61        4
      DO 65 I=1,N3                                                      CY61        5
   65 D(J+I)=A(KPTR+I)                                                  CY61        6
  100 J=J+N3                                                            CNVRT     100
      N=J                                                               CNVRT     101
      DO 105 I=1,N                                                      CNVRT     102
  105 A(I)=D(I)                                                         CNVRT     103
      RETURN                                                            CNVRT     104
  110 CALL ERROR(23)                                                    CNVRT     105
      RETURN                                                            CNVRT     106
      END                                                               CNVRT     107
```

```
      SUBROUTINE COM                                                    COM        2
      COMMON A(1326),D(500),IDTBL(6,500),INITID(3),LASTID(3),ISRCH(3),  RICH       2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A     80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDFS                                  RICH       4
      DIMENSION IDIM(3),IALPH(6)                                        COM        4
      INTEGER SLASH,COMMA,BLANK,A,RPAR                                  COM        5
      INTEGER BITPUT,BITGET                                             COM        6
      DATA (IALPH(I),I=1,6)/1HC,1HO,1HM,1HM,1HO,1HN/                    COM        7
      DATA SLASH/1H//,COMMA/1H,/,BLANK/1H /,RPAR/1H)/,LPAR/1H(/         COM        8
C** COMMON STATEMENT PROCESSOR                                          COM        9
      DO 10 I=1,6                                                       COM       10
      IF(NEXT(JPTR) .NE. IALPH(I)) GO TO 60                             COM       11
   10 CONTINUE                                                          COM       12
      IF(NEXT(JPTR) .EQ. SLASH) GO TO 15                                COM       13
      JPTR=JPTR-1                                                       COM       14
C** BLANK COMMON                                                        COM       15
   12 NXTID=BLANK                                                       COM       16
      GO TO 20                                                          COM       17
C** GET NAME OF LABELLED COMMON BLOCK                                   COM       18
   15 CALL GNLE                                                         COM       19
      IF(A(JPTR-1) .EQ. SLASH) GO TO 12                                 COM       20
      IF(JTYP .NE. 2) GO TO 60                                          COM       21
      IF(NEXT(JPTR) .NE. SLASH) GO TO 60                                COM       22
C** STORE NAME IN SYMBOL TABLE                                          COM       23
   20 CALL COMSCH                                                       COM       24
      IF(ISRCH(3) .EQ. 1) GO TO 25                                      COM       25
      IDTYP=3                                                           COM       26
      CALL STORE                                                        COM       27
      ICMLOC=NID                                                        COM       28
      GO TO 27                                                          COM       29
   25 ICMLOC=LOC                                                        COM       30
C** GET LOCATION OF LAST VARIABLE IN BLOCK                              COM       31
      LSTLOC=IDTBL(6,ICMLOC)                                            COM       32
C** GET NEXT VARIABLE IN BLOCK AND STORE IN SYMBOL TABLE                COM       33
   27 CALL GNLE                                                         COM       34
      IF(JTYP .NE. 2) GO TO 60                                          COM       35
      CALL SEARCH                                                       COM       36
      IF(ISRCH(2) .EQ. 1) CALL ERROR(10,NXTID)                          COM       37
      IF(ISRCH(1) .EQ. 1) GO TO 28                                      COM       38
      IDTYP=1                                                           COM       39
      CALL STORE                                                        COM       40
      LOC=NID                                                           COM       41
C** CHECK VALIDITY AND SET "COMMON" FLAG                                COM       42
   28 IF(BITGET(IDTBL(3,LOC),12,1) .EQ. 1) CALL ERROR(17,NXTID)         COM       43
      IF(BITGET(IDTBL(3,LOC),16,1) .EQ. 1) CALL ERROR(53,NXTID)         COM       44
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),1,16)                            COM       45
      ICMSIZ=1                                                          COM       46
      IF(NEXT(JPTR) .NE. LPAR) GO TO 40                                 COM       47
      IF(BITGET(IDTBL(3,LOC),1,1) .NE. 0) GO TO 80                      COM       48
C** VARIABLE IS DIMENSIONED                                             COM       49
      I=0                                                               COM       50
   35 I=I+1                                                             COM       51
C** GET NEXT DIMENSION AND CHECK SIZE                                   COM       52
      CALL GNLE                                                         COM       53
      IF(JTYP .NE. 5) GO TO 60                                          COM       54
      IDIM(I)=N2                                                        COM       55
      ICMSIZ=ICMSIZ*N2                                                  COM       56
```

```
      IF(N2 .GT. (2**17-1) .OR. N2 .LE. 0) CALL ERROR(8)                COM       57
      IF(NEXT(JPTR) .EQ. COMMA) GO TO 35                                COM       58
      IF(A(JPTR-1) .NE. RPAR) GO TO 60                                  COM       59
      K=NEXT(JPTR)                                                      COM       60
C** SET "DIMENSIONED" FLAG                                              COM       61
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),1,1)                             COM       62
      IF(I .GT. 3) GO TO 60                                             COM       63
C** STORE NO. OF DIMENSIONS AND DIMENSION SIZES IN SYMBOL TABLE          COM       64
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),I,7)                             COM       65
      IF(I-2) 34,32,30                                                  COM       66
   30 IDTBL(4,LOC)=BITPUT(IDTBL(4,LOC),IDIM(3),36)                      COM       67
   32 IDTBL(4,LOC)=BITPUT(IDTBL(4,LOC),IDIM(2),18)                      COM       68
   34 IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),IDIM(1),36)                      COM       69
   40 IF(IDTBL(5,ICMLOC) .EQ. 0) GO TO 45                               COM       70
C** SET POINTER FROM PREVIOUS VARIABLE TO PRESENT VARIABLE IN           COM       71
C** COMMON LINK                                                         COM       72
      IDTBL(5,LSTLOC)=LOC                                               COM       73
      GO TO 47                                                          COM       74
C** FIRST VARIABLE IN BLOCK, STORE LOCATION IN SYMBOL TABLE             COM       75
   45 IDTBL(5,ICMLOC)=LOC                                               COM       76
C** RESET COMMON BLOCK SIZE AND STORE                                   COM       77
   47 IDTBL(4,ICMLOC)=IDTBL(4,ICMLOC)+ICMSIZ                            COM       78
C** RESET "LAST VARIABLE IN BLOCK" TO PRESENT VARIABLE                  COM       79
      IDTBL(6,ICMLOC)=LOC                                               COM       80
C** STORE COMMON LOCATION TO ASSOCIATE THIS VARIABLE                    COM       81
C** WITH THAT COMMON BLOCK                                              COM       82
      IDTBL(6,LOC)=ICMLOC                                               COM       83
      LSTLOC=LOC                                                        COM       84
      IF(A(JPTR-1) .EQ. COMMA) GO TO 27                                 COM       85
      IF(A(JPTR-1) .NE. SLASH) GO TO 50                                 COM       86
C** END OF COMMON BLOCK, COMPLETE COMMON LINK                           COM       87
      IDTBL(5,LOC)=IDTBL(5,ICMLOC)                                      COM       88
      GO TO 15                                                          COM       89
   50 IF(NEXT(JPTR) .NE. BLANK) GO TO 60                                COM       90
C** END OF COMMON BLOCK, COMPLETE COMMON LINK                           COM       91
      IDTBL(5,LOC)=IDTBL(5,ICMLOC)                                      COM       92
      RETURN                                                            COM       93
   60 CALL ERROR(7)                                                     COM       94
      RETURN                                                            COM       95
   80 CALL ERROR(14,NXTID)                                              COM       96
      RETURN                                                            COM       97
      END                                                               COM       98
```

```
      SUBROUTINE COMCHK                                                 COMCHK     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH       2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A     80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH       4
      COMMON/LIST/NLIST,NINTFC,ISUBLT(2,200),INTFAC(300)                COMCHK     4
      COMMON/GLOBAL/NBLK,NREF,NSUBS,BLKTBL(200),EXTTBL(100),ISUBS(100)  COMCHK     5
      INTEGER BITGET,CMBLK(2,20),TP,SZ,PREVTP,BLKTBL                    COMCHK     6
      INTEGER SESCOM(13),SESERR                                         COMCHK     7
      DIMENSION ITPS(6),IORD(6)                                         COMCHK     8
      DATA IORD/1,2,5,4,3,6/                                            COMCHK     9
      DATA SESCOM/4HCASE,3HINA,3HINB,3HINC,3HIOX,5HNPAGX,4HLINX,3HIOY,  COMCHK    10
     $ 5HNPAGY,4HLINY,3HIOZ,5HNPAGZ,4HLINZ/                             COMCHK    11
C** THIS ROUTINE IS CALLED AFTER MODULE PROCESSING TO COMPLETE          COMCHK    12
C** THE PROCESSING OF COMMON BLOCKS                                     COMCHK    13
      SESERR=0                                                          COMCHK    14
      NSES=0                                                            COMCHK    15
      ICTGR2=0                                                          COMCHK    16
      IBLK=INITID(3)                                                    COMCHK    17
      MODCLS=0                                                          COMCHK    18
      LC2=BITGET(IDTBL(3,1),36,9)                                       COMCHK    19
      IF(LC2 .EQ. 0 .OR. IBLKDT .EQ. 1) GO TO 1                         COMCHK    20
      MODCLS=BITGET(ISUBLT(2,LC2),10,4)                                 COMCHK    21
    1 IF(IBLK .EQ. 0) GO TO 120                                         COMCHK    22
      IF(IDTBL(1,IBLK) .EQ. 1H ) GO TO 3                                COMCHK    23
C** GET SESCOMP LIST LOCATION OF COMMON BLOCK                           COMCHK    24
      LISTLC=BITGET(IDTBL(3,IBLK),36,9)                                 COMCHK    25
C** GET COMMON BLOCK CLASS                                              COMCHK    26
      KLAS=BITGET(ISUBLT(2,LISTLC),10,4)                                COMCHK    27
C** GET COMMON BLOCK SIZE                                               COMCHK    28
      ISZ=BITGET(ISUBLT(2,LISTLC),30,15)                                COMCHK    29
C** CHECK SIZE                                                          COMCHK    30
      IF(IDTBL(4,IBLK) .NE. ISZ) GO TO 70                               COMCHK    31
      GO TO 5                                                           COMCHK    32
C** BLANK COMMON                                                        COMCHK    33
    3 IF(MODCLS .NE. 1 .AND. MODCLS .NE. 2) GO TO 5                     COMCHK    34
C** IF CLASS 1 OR 2 - CHECK SIZE                                        COMCHK    35
      IBNKSZ=BITGET(ISUBLT(2,LC2),30,15)                                COMCHK    36
      IF(IDTBL(4,IBLK) .NE. IBNKSZ) CALL ERROR(58,2H//)                 COMCHK    37
    5 NBLOC=0                                                           COMCHK    38
      ISUM=0                                                            COMCHK    39
      NTP=0                                                             COMCHK    40
      TP=0                                                              COMCHK    41
      ICOMST=IDTBL(5,IBLK)                                              COMCHK    42
      LOC=ICOMST                                                        COMCHK    43
   10 PREVTP=TP                                                         COMCHK    44
C** GET TYPE OF NEXT VARIABLE IN COMMON BLOCK                           COMCHK    45
      IF(BITGET(IDTBL(3,LOC),11,1) .EQ. 1) GO TO 15                     COMCHK    46
      TP=1                                                              COMCHK    47
      IFST=BITGET(IDTBL(1,LOC),6,6)                                     COMCHK    48
      IF(IFST .LE. 14 .AND. IFST .GE. 9) TP=4                           COMCHK    49
      GO TO 18                                                          COMCHK    50
   15 TP=BITGET(IDTBL(3,LOC),10,3)                                      COMCHK    51
C** GET SIZE OF VARIABLE                                                COMCHK    52
   18 SZ=1                                                              COMCHK    53
      NDIM=BITGET(IDTBL(3,LOC),7,6)                                     COMCHK    54
      IF(NDIM .EQ. 0) GO TO 22                                          COMCHK    55
      DO 20 I=1,NDIM                                                    COMCHK    56
```

```
      NW=3+(I/2)                                                        COMCHK     57
      ICOL=16*(MOD(I,2)+1)                                              COMCHK     58
   20 SZ=SZ*BITGET(IDTBL(NW,LOC),ICOL,16)                               COMCHK     59
   22 IF(TP .NE. 2 .AND. TP .NE. 3) GO TO 25                            COMCHK     60
C** DOUBLE PRECISION OR COMPLEX VARIABLE                                COMCHK     61
C** CHECK THAT IT BEGINS ON EVEN LOCATION WITHIN COMMON BLOCK            COMCHK     62
      IF(MOD(ISUM,2) .NE. 0)CALL ERROR(64,IDTBL(1,LOC),IDTBL(1,IBLK))   COMCHK     63
      ISUM=ISUM+SZ                                                      COMCHK     64
   25 ISUM=ISUM+SZ                                                      COMCHK     65
C** CHECK FOR PROPER ORDER OF VARIABLES IN BLOCK DATA SUBROUTINE         COMCHK     66
      IF(IBLKDT .EQ. 1 .AND. IORD(PREVTP+1) .GT. IORD(TP+1))            COMCHK     67
     $ CALL ERROR(65,IDTBL(1,IBLK))                                     COMCHK     68
      IF(KLAS .EQ. 10 .OR. IDTBL(1,IBLK) .EQ. 1H ) GO TO 38             COMCHK     69
      IF(KLAS .EQ. 9) GO TO 35                                          COMCHK     70
      IF(KLAS .EQ. 7) GO TO 40                                          COMCHK     71
C** CATEGORY 2 COMMON BLOCK - CHECK THAT IT IS GROUPED BY TYPE          COMCHK     72
      ICTGR2=1                                                          COMCHK     73
      IF(TP .EQ. PREVTP) GO TO 35                                       COMCHK     74
      IF(PREVTP .EQ. 0) GO TO 32                                        COMCHK     75
      DO 30 I=1,NTP                                                     COMCHK     76
      IF(TP .EQ. ITPS(I)) GO TO 110                                     COMCHK     77
   30 CONTINUE                                                          COMCHK     78
   32 NTP=NTP+1                                                         COMCHK     79
      ITPS(NTP)=TP                                                      COMCHK     80
C** CHECK THAT VARIABLE WAS USED                                        COMCHK     81
   35 IF(IBLKDT .EQ. 1) GO TO 38                                        COMCHK     82
      IF(BITGET(IDTBL(3,LOC),38,1) .EQ. 0) CALL ERROR(75,IDTBL(1,LOC))  COMCHK     83
   38 LOC=IDTBL(5,LOC)                                                  COMCHK     84
      IF(LOC .NE. ICOMST) GO TO 10                                      COMCHK     85
      GO TO 65                                                          COMCHK     86
C** CATEGORY 1 COMMON BLOCK - STORE VARIABLE TYPES AND SIZES BY GROUP    COMCHK     87
   40 IF(TP .EQ. PREVTP) GO TO 45                                       COMCHK     88
      NBLOC=NBLOC+1                                                     COMCHK     89
      CMBLK(1,NBLOC)=TP                                                 COMCHK     90
      CMBLK(2,NBLOC)=0                                                  COMCHK     91
   45 CMBLK(2,NBLOC)=CMBLK(2,NBLOC)+SZ                                  COMCHK     92
      IF(IDTBL(1,IBLK) .NE. 6HSESCOM) GO TO 85                          COMCHK     93
C** CHECK VARIABLES IN COMMON BLOCK "SESCOM"                            COMCHK     94
      NSES=NSES+1                                                       COMCHK     95
      IF(NSES .GT. 13) GO TO 80                                         COMCHK     96
      IF(IDTBL(1,LOC) .EQ. SESCOM(NSES)) GO TO 85                       COMCHK     97
   80 SESERR=1                                                          COMCHK     98
   85 LOC=IDTBL(5,LOC)                                                  COMCHK     99
      IF(LOC .NE. ICOMST) GO TO 10                                      COMCHK    100
C** CHECK INTERFACE DEFINITION FOR COMMON BLOCK                         COMCHK    101
C** GET INTERFACE DEFINITION TABLE POINTERS                             COMCHK    102
      IPTR=BITGET(ISUBLT(2,LISTLC),60,12)                               COMCHK    103
      NDPTR=IPTR+(NBLOC-1)/3                                            COMCHK    104
      KOUNT=0                                                           COMCHK    105
      NGRP=NBLOC                                                        COMCHK    106
C** THESE TWO LOOPS CHECK THE INTERFACE DEFINITION                      COMCHK    107
      DO 50 I=IPTR,NDPTR                                                COMCHK    108
      ICOL=0                                                            COMCHK    109
      DO 50 J=1,3                                                       COMCHK    110
      KOUNT=KOUNT+1                                                     COMCHK    111
      IF(KOUNT .GT. NBLOC) GO TO 65                                     COMCHK    112
      ICOL=ICOL+17                                                      COMCHK    113
```

```
C** GET GROUP SIZE                                                    COMCHK    114
      SZ=BITGET(INTFAC(I),ICOL,17)                                    COMCHK    115
      ICOL=ICOL+3                                                     COMCHK    116
C** GET GROUP TYPE                                                    COMCHK    117
      TP=BITGET(INTFAC(I),ICOL,3)                                     COMCHK    118
      IF(TP .NE. 0) GO TO 48                                          COMCHK    119
      CMBLK(2,KOUNT)=CMBLK(2,KOUNT)-SZ                                COMCHK    120
      IF(CMBLK(2,KOUNT) .EQ. 0) GO TO 50                              COMCHK    121
      IF(CMBLK(2,KOUNT) .LT. 0) GO TO 90                              COMCHK    122
      KOUNT=KOUNT-1                                                   COMCHK    123
      NGRP=NGRP+1                                                     COMCHK    124
      GO TO 50                                                        COMCHK    125
C** CHECK INTERFACE DEFINITION FOR SIZE AND TYPE                       COMCHK    126
   48 IF(CMBLK(1,KOUNT) .NE. TP .OR. CMBLK(2,KOUNT) .NE. SZ) GO TO 90 COMCHK    127
   50 CONTINUE                                                        COMCHK    128
      IF(NGRP .NE. BITGET(ISUBLT(2,LISTLC),6,6)) GO TO 90             COMCHK    129
   65 IBLK=IDTBL(2,IBLK)                                              COMCHK    130
      GO TO 1                                                         COMCHK    131
   70 CALL ERROR(58,IDTBL(1,IBLK))                                    COMCHK    132
      GO TO 65                                                        COMCHK    133
   90 CALL ERROR(57,IDTBL(1,IBLK))                                    COMCHK    134
      GO TO 65                                                        COMCHK    135
  110 CALL ERROR(63,IDTBL(1,IBLK))                                    COMCHK    136
      GO TO 65                                                        COMCHK    137
C** CHECK THAT COMMON BLOCK "SESCOM" IS WELL DEFINED                  COMCHK    138
  120 IF(ICTGR2 .EQ. 0 .AND. (MODCLS .EQ. 1 .OR. MODCLS .EQ. 2))      COMCHK    139
     $ CALL ERROR(73)                                                 COMCHK    140
      IF(NSES .EQ. 0) GO TO 130                                       COMCHK    141
      IF(SESERR .EQ. 1 .OR. NSES .LT. 13) CALL ERROR(48)              COMCHK    142
      RETURN                                                          COMCHK    143
  130 CALL ERROR(66)                                                  COMCHK    144
      RETURN                                                          COMCHK    145
      END                                                             COMCHK    146
```

```
      SUBROUTINE COMEXT                                                 COMEXT     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH      2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A     80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IOFS                                  RICH      4
      INTEGER BITGET                                                    COMEXT     4
C** THIS ROUTINE CHECKS TO SEE IF AN EQUIVALENCE STATEMENT              COMEXT     5
C** EXTENDS COMMON                                                      COMEXT     6
      ICOMLC=0                                                          COMEXT     7
C** GET COMMON INFORMATION FROM SYMBOL TABLE                            COMEXT     8
      ICOMNM=IDTBL(6,LOC)                                               COMEXT     9
      ICOMST=IDTBL(5,ICOMNM)                                            COMEXT    10
      ICOMND=IDTBL(6,ICOMNM)                                            COMEXT    11
      ICOMSZ=IDTBL(4,ICOMNM)                                            COMEXT    12
      NXTLOC=ICOMND                                                     COMEXT    13
C** THIS LOOP COMPUTES THE NUMBER OF COMMON LOCATIONS TO THE LEFT OF    COMEXT    14
C** THE EQUIVALENCED VARIABLE                                          COMEXT    15
      DO 20 I=1,ICOMSZ                                                  COMEXT    16
      NXTLOC=IDTBL(5,NXTLOC)                                            COMEXT    17
      MUL=1                                                             COMEXT    18
      ISZ=1                                                             COMEXT    19
C** GET VARIABLE TYPE                                                  COMEXT    20
      ITP=BITGET(IDTBL(3,NXTLOC),10,3)                                  COMEXT    21
C** IF COMPLEX OR DOUBLE PRECISION, SET MULTIPLIER TO 2                 COMEXT    22
      IF(ITP .EQ. 2 .OR. ITP .EQ. 3) MUL=2                              COMEXT    23
      IF(LOC .EQ. NXTLOC) GO TO 25                                      COMEXT    24
      IF(BITGET(IDTBL(3,NXTLOC),1,1) .EQ. 1) GO TO 10                   COMEXT    25
C** NON-DIMENSIONED VARIABLE                                           COMEXT    26
      GO TO 20                                                          COMEXT    27
   10 NDIM=BITGET(IDTBL(3,NXTLOC),7,6)                                  COMEXT    28
C** DIMENSIONED VARIABLE - GET SIZE OF ARRAY                           COMEXT    29
      DO 15 J=1,NDIM                                                    COMEXT    30
      IWRD=3+J/2                                                        COMEXT    31
      IPOS=(MOD(J,2)+1)*18                                              COMEXT    32
      ISZ=ISZ*BITGET(IDTBL(IWRD,NXTLOC),IPOS,18)                        COMEXT    33
   15 CONTINUE                                                          COMEXT    34
   20 ICOMLC=ICOMLC+ISZ*MUL                                             COMEXT    35
   25 ILFT=ICOMLC                                                       COMEXT    36
C** COMPUTE NUMBER OF COMMON LOCATIONS TO THE RIGHT OF THE              COMEXT    37
C** EQUIVALENCED VARIABLE                                              COMEXT    38
      IRHT=ICOMSZ-ICOMLC                                                COMEXT    39
      NXTLOC=LOC                                                        COMEXT    40
C** GET OFFSET OF EQUIVALENCED VARIABLE                                 COMEXT    41
      IOFFST=IDTBL(8,NXTLOC)                                            COMEXT    42
   30 NXTLOC=IDTBL(7,NXTLOC)                                            COMEXT    43
      IF(NXTLOC .EQ. LOC) RETURN                                        COMEXT    44
C** GET OFFSET OF NEXT VARIABLE IN EQUIVALENCE LINK                     COMEXT    45
      IOFF2=IDTBL(8,NXTLOC)                                             COMEXT    46
C** CHECK TO SEE IF COMMON WAS EXTENDED TO THE LEFT                     COMEXT    47
      IF((IOFFST-IOFF2) .GT. ILFT) GO TO 50                             COMEXT    48
      ITP=BITGET(IDTBL(3,NXTLOC),10,3)                                  COMEXT    49
      ISZ=1                                                             COMEXT    50
      IF(ITP .NE. 2 .AND. ITP .NE. 3) GO TO 32                          COMEXT    51
C** IF VARIABLE IS COMPLEX OR DOUBLE PRECISION, CHECK THAT IT BEGINS    COMEXT    52
C** ON AN EVEN LOCATION WITHIN COMMON BLOCK                            COMEXT    53
      IF(MOD((ILFT-IOFFST+IOFF2),2).NE. 0)CALL ERROR(64,IDTBL(1,NXTLOC)) COMEXT    54
      ISZ=2                                                             COMEXT    55
   32 IF(BITGET(IDTBL(3,NXTLOC),1,1) .NE. 1) GO TO 40                   COMEXT    56
```

```
      NDIM=BITGET(IDTBL(3,NXTLOC),7,6)                                  COMEXT     57
C** DIMENSIONED VARIABLE - GET SIZE OF ARRAY                            COMEXT     58
      DO 35 I=1,NDIM                                                    COMEXT     59
      IWRD=3+I/2                                                        COMEXT     60
      IPOS=(MOD(I,2)+1)*18                                              COMEXT     61
      ISUB=BITGET(IDTBL(IWRD,NXTLOC),IPOS,18)                           COMEXT     62
      ISZ=ISZ*ISUB                                                      COMEXT     63
   35 CONTINUE                                                          COMEXT     64
C** CHECK TO SEE IF COMMON WAS EXTENDED TO THE RIGHT                    COMEXT     65
   40 IF((ISZ-(IOFFST-IOFF2)) .GT. IRHT) GO TO 50                       COMEXT     66
      GO TO 30                                                          COMEXT     67
   50 CALL ERROR(47)                                                    COMEXT     68
      RETURN                                                            COMEXT     69
      END                                                               COMEXT     70
```

```
      SUBROUTINE COMSCH                                                 COMSCH      2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH        2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A      80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH        4
C** THIS ROUTINE SEARCHES THE SYMBOL TABLE FOR A COMMON BLOCK NAME      COMSCH      4
C** AND RETURNS        ISRCH(3)=1 - FOUND       =0 - NOT FOUND         COMSCH      5
C**                    LOC - SYMBOL TABLE LOCATION WHERE NAME WAS FOUND COMSCH      6
      J=INITID(3)                                                       COMSCH      7
      IF(J .EQ. 0) GO TO 15                                             COMSCH      8
      DO 10 I=1,NID                                                     COMSCH      9
      IF(IDTBL(1,J) .NE. NXTID) GO TO 5                                 COMSCH     10
      ISRCH(3)=1                                                        COMSCH     11
      LOC=J                                                             COMSCH     12
      RETURN                                                            COMSCH     13
    5 J=IDTBL(2,J)                                                      COMSCH     14
      IF(J .EQ. 0) GO TO 15                                             COMSCH     15
   10 CONTINUE                                                          COMSCH     16
   15 ISRCH(3) =0                                                       COMSCH     17
      RETURN                                                            COMSCH     18
      END                                                               COMSCH     19
```

```
      SUBROUTINE CTGOTO                                                 CTGOTO     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH       2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A     80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH       4
      COMMON/LABELS/STATRA(2,200),NLABEL                                CTGOTO     4
      COMMON/BASBLK/IBLOCK(2500),NBLOCK,NB,NBRNCH                       CY58A     28
      DIMENSION IALPH(4)                                                CTGOTO     6
      INTEGER STATRA,A,BLANK,RPAR,COMMA                                 CTGOTO     7
      INTEGER BITPUT,BITGET                                             CTGOTO     8
      DATA BLANK/1H /,COMMA/1H,/,LPAR/1H(/,RPAR/1H)/                    CTGOTO     9
      DATA (IALPH(I),I=1,4)/1HG,1HO,1HT,1HO/                            CTGOTO    10
C** COMPUTED GO TO STATEMENT PROCESSOR                                  CTGOTO    11
      DO 5 I=1,4                                                        CTGOTO    12
      IF(NEXT(JPTR) .NE. IALPH(I)) GO TO 30                             CTGOTO    13
    5 CONTINUE                                                          CTGOTO    14
      IF(NEXT(JPTR) .NE. LPAR) GO TO 30                                 CTGOTO    15
      NBLOCK=NBLOCK+1                                                   CTGOTO    16
      JBLOCK=NBLOCK                                                     CTGOTO    17
      NBRNCH=0                                                          CTGOTO    18
C** GET NEXT STATEMENT LABEL                                            CTGOTO    19
   10 CALL GNLE                                                         CTGOTO    20
      IF(JTYP .NE. 5) GO TO 30                                          CTGOTO    21
C** SEARCH STATEMENT NUMBER TABLE AND SET "GOTO" FLAG                   CTGOTO    22
      CALL STSRCH                                                       CTGOTO    23
      STATRA(2,LOC)=BITPUT(STATRA(2,LOC),1,12)                          CTGOTO    24
      IF(NBRNCH .EQ. 0) GO TO 15                                        CTGOTO    25
C** CHECK FOR POSSIBLE DUPLICATE BRANCHES                               CTGOTO    26
      DO 12 I=1,NBRNCH                                                  CTGOTO    27
      IF(LOC .EQ. IBLOCK(NBLOCK-I+1)) GO TO 17                          CTGOTO    28
   12 CONTINUE                                                          CTGOTO    29
C** STORE BRANCH IN BASIC BLOCK TABLE                                   CTGOTO    30
   15 NBLOCK=NBLOCK+1                                                   CTGOTO    31
      IBLOCK(NBLOCK)=LOC                                                CTGOTO    32
C** INCREMENT BRANCH COUNTER                                            CTGOTO    33
      NBRNCH=NBRNCH+1                                                   CTGOTO    34
   17 IF(NEXT(JPTR) .EQ. COMMA) GO TO 10                                CTGOTO    35
      IF(A(JPTR-1) .NE. RPAR) GO TO 30                                  CTGOTO    36
      IF(NEXT(JPTR) .NE. COMMA) GO TO 30                                CTGOTO    37
C** GET VARIABLE REFERENCE                                              CTGOTO    38
      CALL GNLE                                                         CTGOTO    39
      IF(JTYP .NE. 2) GO TO 30                                          CTGOTO    40
C** GET SYMBOL TABLE LOCATION                                           CTGOTO    41
      CALL SEARCH                                                       CTGOTO    42
      IF(ISRCH(2) .EQ. 1) CALL ERROR(10,NXTID)                          CTGOTO    43
      IF(ISRCH(1) .EQ. 1) GO TO 20                                      CTGOTO    44
      IDTYP=1                                                           CTGOTO    45
      CALL STORE                                                        CTGOTO    46
      LOC=NID                                                           CTGOTO    47
C** CHECK THAT REFERENCE IS INTEGER VARIABLE                            CTGOTO    48
   20 CALL IMPTYP                                                       CTGOTO    49
      IF(BITGET(IDTBL(3,LOC),10,3) .NE. 4) CALL ERROR(39,NXTID)         CTGOTO    50
      IF(BITGET(IDTBL(3,LOC),1,1) .EQ. 1) CALL ERROR(14,NXTID)          CTGOTO    51
      IF(NEXT(JPTR) .NE. BLANK) GO TO 30                                CTGOTO    52
C** STORE REFERENCE IN BASIC BLOCK TABLE                                CTGOTO    53
      IBLOCK(JBLOCK)=2000+LOC                                           CTGOTO    54
      NB=1                                                              CTGOTO    55
      RETURN                                                            CTGOTO    56
   30 CALL ERROR(7)                                                     CTGOTO    57
      RETURN                                                            CTGOTO    58
      END                                                               CTGOTO    59
```

```
      SUBROUTINE DATA                                                    DATA      2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),   RICH      2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,         CY58A    80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                   RICH      4
      DIMENSION IALPH(4)                                                 DATA      4
      INTEGER A,RPAR,COMMA,SLASH,BLANK,ASTRIK,PLUS                       DATA      5
      INTEGER BITPUT,BITGET                                              DATA      6
      DATA LPAR/1H(/,RPAR/1H)/,COMMA/1H,/,SLASH/1H//,BLANK/1H /,         DATA      7
     1   ASTRIK/1H*/,PLUS/1H+/,MINUS/1H-/                                DATA      8
      DATA (IALPH(I),I=1,4)/1HD,1HA,1HT,1HA/                             DATA      9
C** DATA STATEMENT PROCESSOR                                             DATA     10
      DO 5 I=1,4                                                         DATA     11
      IF(NEXT(JPTR) .NE. IALPH(I)) GO TO 60                              DATA     12
    5 CONTINUE                                                           DATA     13
    6 LST1SZ=0                                                           DATA     14
      LST2SZ=0                                                           DATA     15
    8 ISZ=1                                                              DATA     16
C** GET NEXT VARIABLE NAME                                               DATA     17
      CALL GNLE                                                          DATA     18
      IF(JTYP .NE. 2) GO TO 60                                           DATA     19
C** STORE IN SYMBOL TABLE                                                DATA     20
      CALL SEARCH                                                        DATA     21
      IF(ISRCH(2) .EQ. 1) CALL ERROR(10,NXTID)                           DATA     22
      IF(ISRCH(1) .EQ. 1) GO TO 9                                        DATA     23
      IDTYP=1                                                            DATA     24
      CALL STORE                                                         DATA     25
      LOC=NID                                                            DATA     26
    9 IF(BITGET(IDTBL(3,LOC),12,1) .EQ. 1) CALL ERROR(30,NXTID)          DATA     27
C** SET TYPE                                                             DATA     28
      CALL IMPTYP                                                        DATA     29
      IF(BITGET(IDTBL(3,LOC),16,1) .EQ. 0) GOTO 10                       DATA     30
C** VARIABLE IN COMMON - MUST BE BLOCK DATA SUBPROGRAM                   DATA     31
C** MUST ALSO BE LABELLED COMMON                                         DATA     32
      ICOMLC=IDTBL(6,LOC)                                                DATA     33
      IF(IBLKDT .EQ. 0 .OR. IDTBL(1,ICOMLC) .EQ. BLANK) CALL ERROR(28,   DATA     34
     *NXTID)                                                             DATA     35
   10 IF(NEXT(JPTR) .NE. LPAR) GO TO 25                                  DATA     36
      IF(BITGET(IDTBL(3,LOC),1,1) .EQ. 0) GO TO 90                       DATA     37
C** SUBSCRIPTED VARIABLE - GET NO. OF DIMENSIONS                         DATA     38
      NDIM=BITGET(IDTBL(3,LOC),7,6)                                      DATA     39
C** THIS LOOP CHECKS THE SUBSCRIPTS AGAINST THE ACTUAL DIMENSIONS        DATA     40
C** TO CHECK THEIR VALIDITY                                              DATA     41
      I=0                                                                DATA     42
   15 I=I+1                                                              DATA     43
      CALL GNLE                                                          DATA     44
      IF(JTYP .NE. 5) GO TO 60                                           DATA     45
      IF(N2 .LE. 0) CALL ERROR(8)                                        DATA     46
      IWRD=3+I/2                                                         DATA     47
      IPOS=18*MOD(I,2)+18                                                DATA     48
      IF(N2 .GT. BITGET(IDTBL(IWRD, LOC),IPOS,18)) CALL ERROR(18)        DATA     49
      IF(NEXT(JPTR) .EQ. COMMA) GO TO 15                                 DATA     50
      IF(I .NE. NDIM) GO TO 80                                           DATA     51
      IF(A(JPTR-1) .NE. RPAR) GO TO 80                                   DATA     52
      GO TO 35                                                           DATA     53
C** NON-SUBSCRIPTED VARIABLE                                             DATA     54
   25 JPTR=JPTR-1                                                        DATA     55
      IF(BITGET(IDTBL(3,LOC),1,1) .EQ. 0) GO TO 35                       DATA     56
```

```
C** ARRAY NAME                                                          DATA       57
      NDIM=BITGET(IDTBL(3,LOC),7,6)                                     DATA       58
C** THIS LOOP CALCULATES THE SIZE OF THE ARRAY                          DATA       59
      DO 30 I=1,NDIM                                                    DATA       60
      IWRD=3+I/2                                                        DATA       61
      IPOS=18*MOD(I,2)+18                                               DATA       62
   30 ISZ=ISZ*BITGET(IDTBL(IWRD,LOC),IPOS,18)                           DATA       63
      IF(BITGET(IDTBL(3,LOC),14,1) .EQ. 1) CALL ERROR(29,IDTBL(1,LOC))  DATA       64
C** INCREMENT VARIABLE LIST SIZE BY APPROPRIATE AMOUNT                  DATA       65
   35 LST1SZ=LST1SZ+ISZ                                                 DATA       66
C** SET "DATA" FLAG                                                     DATA       67
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),1,14)                            DATA       68
      IF(NEXT(JPTR) .EQ. COMMA) GO TO 8                                 DATA       69
      IF(A(JPTR-1) .NE. SLASH) GO TO 60                                 DATA       70
C** SLASH ENCOUNTERED - END OF VARIABLE LIST                            DATA       71
   40 NRPEAT=1                                                          DATA       72
      CALL GNLE                                                         DATA       73
C** GET NEXT CONSTANT                                                   DATA       74
      IF(JTYP .EQ. 3) GO TO 47                                          DATA       75
      IF(JTYP .NE. 5) GO  TO 45                                         DATA       76
C** INTEGER - MAY BE A REPEAT COUNT                                     DATA       77
      IF(NEXT(JPTR) .NE. ASTRIK) GO TO 50                               DATA       78
C** REPEAT COUNT - SET VALUE                                            DATA       79
      NRPEAT=N2                                                         DATA       80
C** GET NEXT CONSTANT                                                   DATA       81
      CALL GNLE                                                         DATA       82
C** CONSTANT MAY BE PRECEDED BY PLUS OR MINUS                           DATA       83
   45 IF(A(JPTR-1) .NE. PLUS .AND. A(JPTR-1) .NE. MINUS) GO TO 47       DATA       84
C** PLUS OR MINUS FOUND, GET NEXT CHARACTER                             DATA       85
      CALL GNLE                                                         DATA       86
   47 KK=NEXT(JPTR)                                                     DATA       87
   50 IF(JTYP .GE. 3.AND. JTYP .LE. 6) GO TO 55                         DATA       88
      IF(JTYP .EQ. 7 .AND. (LOGID .EQ. 10 .OR. LOGID .EQ. 11)) GO TO 35 DATA       89
      GO TO 70                                                          DATA       90
   55 LST2SZ=LST2SZ+NRPEAT                                              DATA       91
C** INCREMENT CONSTANT LIST SIZE                                        DATA       92
      IF(A(JPTR-1) .EQ. COMMA) GO TO 40                                 DATA       93
      IF(A(JPTR-1) .NE. SLASH) GO TO 60                                 DATA       94
C** COMPARE SIZE OF CONSTANT LIST WITH SIZE OF VARIABLE LIST            DATA       95
      IF(LST1SZ .NE. LST2SZ) CALL ERROR(31)                             DATA       96
      IF(NEXT(JPTR) .EQ. COMMA) GO TO 6                                 DATA       97
      IF(A(JPTR-1) .NE. BLANK) GO TO 60                                 DATA       98
      RETURN                                                            DATA       99
   60 CALL ERROR(7)                                                     DATA      100
      RETURN                                                            DATA      101
   70 CALL ERROR(23)                                                    DATA      102
      RETURN                                                            DATA      103
   80 CALL ERROR(19)                                                    DATA      104
      RETURN                                                            DATA      105
   90 CALL ERROR(13,IDTBL(1,LOC))                                       DATA      106
      RETURN                                                            DATA      107
      END                                                               DATA      108
```

```
      SUBROUTINE DESCRP                                                 DESCRP     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH      2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A     80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH      4
      COMMON/FORMAT/IDESST,IDESND,IGPST,IGPND,IGRP,SEPST,SEPND,         CY58A     5
     1 DIR,ICOM,ISEP                                                    DESCRP    5
      DIMENSION FORMT(7)                                                DESCRP    6
      INTEGER A,FORMT,DECPT,BLANK,PEE,EX,AICH                           DESCRP    7
      DATA (FORMT(I),I=1,7)/1HF,1HE,1HG,1HD,1HI,1HL,1HA/                DESCRP    8
      DATA DECPT/1H./,BLANK/1H /,PEE/1HP/,EX/1HX/,AICH/1HH/,MINUS/1H-/  DESCRP    9
C** THIS ROUTINE CHECKS THE SYNTAX OF A FIELD DESCRIPTOR AND RETURNS    DESCRP    10
C**      IDES=1 - VALID                                                 DESCRP    11
C**      IDES=0 - INVALID                                               DESCRP    12
      ISCLFC=0                                                          DESCRP    13
      INT=0                                                             DESCRP    14
      IMINUS=0                                                          DESCRP    15
      IDES=1                                                            DESCRP    16
      IF(NEXT(IDESST) .NE. MINUS) GO TO 5                               DESCRP    17
C** MINUS SIGN FOUND, SCALE FACTOR SHOULD FOLLOW                        DESCRP    18
      IMINUS=1                                                          DESCRP    19
      GO TO 6                                                           DESCRP    20
    5 JPTR=IDESST                                                       DESCRP    21
    6 CONTINUE                                                          DESCRP    22
      CALL GNLE                                                         DESCRP    23
      IF(JTYP .EQ. 3) GO TO 80                                          DESCRP    24
      IF(JTYP .EQ. 5) GO TO 10                                          DESCRP    25
      IF(JTYP .NE. 2) GO TO 15                                          DESCRP    26
      IF(NXTID .EQ. PEE .AND. ISCLFC .EQ. 0) GO TO 20                   DESCRP    27
      IF(INT .EQ. 1 .AND. N2 .LT. 1) GO TO 15                           DESCRP    28
      IF(ISCLFC .EQ. 0 .AND. IMINUS .EQ. 1) GO TO 15                    DESCRP    29
      GO TO 25                                                          DESCRP    30
   10 INT=1                                                             DESCRP    31
      GO TO 6                                                           DESCRP    32
C** INTEGER FOUND                                                       DESCRP    33
   15 IDES=0                                                            DESCRP    34
      RETURN                                                            DESCRP    35
   20 IF(INT .EQ. 0) GO TO 15                                           DESCRP    36
C** SCALE FACTOR FOUND                                                  DESCRP    37
      ISCLFC=1                                                          DESCRP    38
      INT=0                                                             DESCRP    39
      GO TO 6                                                           DESCRP    40
C** GET TYPE OF FIELD DESCRIPTOR                                        DESCRP    41
   25 DO 30 I=1,7                                                       DESCRP    42
      IF(NXTID .EQ. FORMT(I)) GO TO 45                                  DESCRP    43
   30 CONTINUE                                                          DESCRP    44
      IF(NXTID .NE. EX .OR. ISCLFC .EQ. 1 .OR. INT .EQ. 0) GO TO 15     DESCRP    45
      GO TO 80                                                          DESCRP    46
   45 CALL GNLE                                                         DESCRP    47
C** GET FIELD WIDTH                                                     DESCRP    48
      IF(JTYP .NE. 5) GO TO 15                                          DESCRP    49
      NWIDTH=N2                                                         DESCRP    50
      IF(I .LE. 4) GO TO 60                                             DESCRP    51
C** FIELD TYPE IS I,L, OR A                                             DESCRP    52
      IF(ISCLFC .EQ. 1 .OR. NWIDTH .LT. 1 .OR. (I .EQ. 7 .AND. NWIDTH   DESCRP    53
     $ .GT. 4)) GO TO 15                                                DESCRP    54
      IDESND=JPTR-1                                                     DESCRP    55
      RETURN                                                            DESCRP    56
C** FIELD TYPE IS F,E,G OR D                                            DESCRP    57
   60 IF(NEXT(JPTR) .NE. DECPT) GO TO 15                                DESCRP    58
      IF(NWIDTH .LT. 2) GO TO 15                                        DESCRP    59
      CALL GNLE                                                         DESCRP    60
      IF(JTYP .NE. 5) GO TO 15                                          DESCRP    61
C** GET NUMBER OF DECIMAL PLACES                                        DESCRP    62
      NDCPLS=N2                                                         DESCRP    63
      IDESND=JPTR-1                                                     DESCRP    64
      IF(I .EQ. 1) GO TO 65                                             DESCRP    65
C** CHECK VALIDITY OF FIELD DESCRIPTOR SIZE                             DESCRP    66
      IF(NWIDTH .LT. (NDCPLS+6)) GO TO 15                               DESCRP    67
      RETURN                                                            DESCRP    68
   65 IF(NWIDTH .LT. NDCPLS) GO TO 15                                   DESCRP    69
      RETURN                                                            DESCRP    70
   80 IDESND=JPTR-1                                                     DESCRP    71
      RETURN                                                            DESCRP    72
      END                                                               DESCRP    73
```

```
      SUBROUTINE DIMEN                                                  DIMEN     2
      COMMON A(13261),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH     2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A    80
     2 LTYP,ITYP,IBLKOT,MODE,IERR,IDES                                  RICH      4
      DIMENSION IALPH(9),IDIM(3)                                        DIMEN     4
      INTEGER A,D,RPAR,COMMA                                            DIMEN     5
      INTEGER BITPUT,BITGET,COMLOC                                      DIMEN     6
      DATA (IALPH(I),I=1,9)/1HD,1HI,1HM,1HE,1HN,1HS,1HI,1HO,1HN/        DIMEN     7
      DATA LPAR/1H(/,RPAR/1H)/,COMMA/1H,/                               DIMEN     8
C** DIMENSION STATEMENT PROCESSOR                                       DIMEN     9
C** CHECK SPELLING                                                      DIMEN    10
      DO 10 I=1,9                                                       DIMEN    11
      IF(NEXT(JPTR) .NE. IALPH(I)) GO TO 110                            DIMEN    12
   10 CONTINUE                                                          DIMEN    13
C** GET NEXT DIMENSIONED VARIABLE AND STORE IN SYMBOL TABLE             DIMEN    14
   12 CALL GNLE                                                         DIMEN    15
      IF(JTYP .NE. 2) GO TO 110                                         DIMEN    16
      CALL SEARCH                                                       DIMEN    17
      IF(ISRCH(2) .EQ. 1) CALL ERROR(10,NXTID)                          DIMEN    18
      IF(ISRCH(1) .EQ. 1) GO TO 5                                       DIMEN    19
      IDTYP=1                                                           DIMEN    20
      CALL STORE                                                        DIMEN    21
      LOC=NID                                                           DIMEN    22
C** IF PREVIOUSLY DIMENSIONED, ISSUE DIAGNOSTIC                         DIMEN    23
    5 IF(BITGET(IDTBL(3,LOC),1,1) .NE. 0) CALL ERROR(11,NXTID)          DIMEN    24
C** SET TYPE                                                            DIMEN    25
      CALL IMPTYP                                                       DIMEN    26
C** SET "DIMENSIONED" FLAG                                              DIMEN    27
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),1,1)                             DIMEN    28
      IE=LOC                                                            DIMEN    29
      IF(NEXT(JPTR) .NE. LPAR) GO TO 110                                DIMEN    30
      INCR=1                                                            DIMEN    31
      I=0                                                               DIMEN    32
   15 I=I+1                                                             DIMEN    33
C** GET NEXT DIMENSION                                                  DIMEN    34
      CALL GNLE                                                         DIMEN    35
      IF(JTYP .NE. 5) GO TO 13                                          DIMEN    36
C** DIMENSION IS A CONSTANT, CHECK SIZE                                 DIMEN    37
      IDIM(I)=N2                                                        DIMEN    38
      IF(N2 .GT. (2**17-1) .OR. N2 .LE. 0) CALL ERROR(8)                DIMEN    39
      INCR=INCR*N2                                                      DIMEN    40
      GO TO 14                                                          DIMEN    41
   13 IF(JTYP .NE. 2) GO TO 110                                         DIMEN    42
C** VARIABLE DIMENSION, STORE IN SYMBOL TABLE AND CHECK VALIDITY        DIMEN    43
      IDTYP=1                                                           DIMEN    44
      CALL SEARCH                                                       DIMEN    45
      IF(ISRCH(2) .NE. 0) CALL ERROR(10,NXTID)                          DIMEN    46
      IF(ISRCH(1) .EQ. 1) GOTO 25                                       DIMEN    47
      IDTYP=1                                                           DIMEN    48
      CALL STORE                                                        DIMEN    49
      LOC=NID                                                           DIMEN    50
   25 IF(BITGET(IDTBL(3,LOC),12,1) .NE. 1) CALL ERROR(9)                DIMEN    51
      IF(BITGET(IDTBL(3,LOC),1,1) .NE. 0) GO TO 120                     DIMEN    52
C** SET "VARIABLE DIMENSION" FLAG                                       DIMEN    53
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),1,13)                            DIMEN    54
C** SET TYPE AND MAKE SURE IT IS INTEGER                                DIMEN    55
      CALL IMPTYP                                                       DIMEN    56
```

```
      IF(BITGET(IDTBL(3,LOC),10,3) .NE. 4) CALL ERROR(9)                DIMEN    57
      IDIM(I)=2**17*LOC                                                 DIMEN    58
   14 IF(NEXT(JPTR) .EQ. COMMA) GO TO 15                                DIMEN    59
      IF(A(JPTR-1) .NE. RPAR) GO TO 110                                 DIMEN    60
      LOC=IE                                                            DIMEN    61
C** STORE NO. OF DIMENSIONS                                             DIMEN    62
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),I,7)                             DIMEN    63
      IF(I .GT. 3) GO TO 110                                            DIMEN    64
      IF(I-2) 35,30,24                                                  DIMEN    65
   24 IDTBL(4,LOC)=BITPUT(IDTBL(4,LOC),IDIM(3),36)                      DIMEN    66
C** STORE DIMENSION SIZES                                               DIMEN    67
   30 IDTBL(4,LOC)=BITPUT(IDTBL(4,LOC),IDIM(2),18)                      DIMEN    68
   35 IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),IDIM(1),36)                      DIMEN    69
      IF(BITGET(IDTBL(3,LOC),16,1) .NE. 1) GO TO 50                     DIMEN    70
C** VARIABLE IN COMMON, RESET COMMON BLOCK SIZE                         DIMEN    71
      COMLOC=IDTBL(6,LOC)                                               DIMEN    72
      IT=1                                                              DIMEN    73
      ITP=BITGET(IDTBL(3,LOC),10,3)                                     DIMEN    74
      IF(ITP .EQ. 2 .OR. ITP .EQ. 3) IT=2                               DIMEN    75
      IDTBL(4,COMLOC)=IDTBL(4,COMLOC)+IT*(INCR-1)                       DIMEN    76
   50 CONTINUE                                                          DIMEN    77
      IF(NEXT(JPTR) .EQ. COMMA) GO TO 12                                DIMEN    78
      IF(JPTR .GT. N) RETURN                                            DIMEN    79
  110 CALL ERROR(7)                                                     DIMEN    80
      RETURN                                                            DIMEN    81
  120 CALL ERROR(14,NXTID)                                              DIMEN    82
      RETURN                                                            DIMEN    83
      END                                                               DIMEN    84
```

```
      SUBROUTINE DO                                                    DO          2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH        2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,       CY58A      80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                 RICH        4
      COMMON/LABELS/STATRA(2,200),NLABEL                               DO          4
      COMMON/DOLOOP/ISTACK(4,50),NSTACK,ILOOP,IOVFLW                   DO          5
      COMMON/BASBLK/IBLOCK(2500),NBLOCK,NB,NBRNCH                      CY58A      31
      DIMENSION PARAM(3)                                               DO          7
      INTEGER A,BLANK,COMMA,EQUALS,DEE,OH,PARAM,STATRA                 DO          8
      INTEGER BITPUT,BITGET                                            DO          9
      DATA BLANK/1H /,COMMA/1H,/,EQUALS/1H=/,DEE/1HD/,OH/1HO/          DO         10
C** DO STATEMENT PROCESSOR                                             DO         11
      IF(NEXT(JPTR) .NE. DEE) GO TO 50                                 DO         12
      IF(NEXT(JPTR) .NE. OH) GO TO 50                                  DO         13
C** GET DO TERMINAL                                                    DO         14
      CALL GNLE                                                        DO         15
      IF(JTYP .NE. 5) GOTO 50                                          DO         16
C** SEARCH STATEMENT NUMBER TABLE                                      DO         17
      CALL STSRCH                                                      DO         18
C** SET FLAGS                                                          DO         19
      STATRA(2,LOC)=BITPUT(STATRA(2,LOC),1,12)                         DO         20
      STATRA(2,LOC)=BITPUT(STATRA(2,LOC),1,15)                         DO         21
      IF(IOVFLW .EQ. 1) GO TO 2                                        DO         22
C** INCREMENT DO STACK COUNTER                                         DO         23
      NSTACK=NSTACK+1                                                  DO         24
      IF(NSTACK .GT. 50) GO TO 1                                       DO         25
C** STORE DO TERMINAL AND "CURRENT LOOP" IN DO STACK                   DO         26
      ISTACK(1,NSTACK)=LOC                                             DO         27
      ISTACK(2,NSTACK)=0                                               DO         28
      ISTACK(3,NSTACK)=ILOOP                                           DO         29
C** RESET VALUE OF CURRENT LOOP                                        DO         30
      ILOOP=NSTACK                                                     DO         31
      GO TO 2                                                          DO         32
    1 IOVFLW=1                                                         DO         33
      WRITE(6,60)                                                      DO         34
   60 FORMAT(///5X,50H DO STACK OVERFLOW - DO LOOP PROCESSING TERMINATED DO         35
     *)                                                                DO         36
C** GET DO INDEX                                                       DO         37
    2 CALL GNLE                                                        DO         38
      IF(JTYP .NE. 2) GO TO 50                                         DO         39
C** GET SYMBOL TABLE LOCATION                                          DO         40
      CALL SEARCH                                                      DO         41
      IF(ISRCH(2) .EQ. 1) CALL ERROR(10,NXTID)                         DO         42
      IF(ISRCH(1) .EQ. 1) GO TO 5                                      DO         43
      IDTYP=1                                                          DO         44
      CALL STORE                                                       DO         45
      LOC=NID                                                          DO         46
C** CHECK THAT INDEX IS AN INTEGER VARIABLE                            DO         47
    5 CALL IMPTYP                                                      DO         48
      IF(BITGET(IDTBL(3,LOC),10,3) .NE. 4) CALL ERROR(40,NXTID)        DO         49
      IF(BITGET(IDTBL(3,LOC),1,1) .EQ. 1) CALL ERROR(14,NXTID)         DO         50
      IF(NEXT(JPTR) .NE. EQUALS) GO TO 50                              DO         51
      IF(IOVFLW .EQ. 1) GO TO 8                                        DO         52
C** STORE DO INDEX IN BASIC BLOCK TABLE                                DO         53
      NBLOCK=NBLOCK+1                                                  DO         54
      IBLOCK(NBLOCK)=3000+LOC                                          DO         55
C** STORE DO INDEX IN DO STACK                                         DO         56
```

```
      ISTACK(4,NSTACK)=LOC                                              DO        57
    8 PARAM(3)=1                                                        DO        58
C** GET NEXT DO PARAMETER                                               DO        59
      DO 30 I=1,3                                                       DO        60
      CALL GNLE                                                         DO        61
      IF(JTYP .NE. 5) GO TO 10                                          DO        62
C** DO PARAMETER IS AN INTEGER - STORE VALUE                            DO        63
      PARAM(I)=N2                                                       DO        64
      IF(N2 .LE. 0) CALL ERROR(41)                                      DO        65
      GO TO 20                                                          DO        66
   10 IF(JTYP .NE. 2) GO TO 50                                          DO        67
C** DO PARAMETER IS A VARIABLE - GET SYMBOL TABLE LOCATION               DO        68
      CALL SEARCH                                                       DO        69
      IF(ISRCH(2) .EQ. 1) CALL ERROR(10,NXTID)                          DO        70
      IF(ISRCH(1) .EQ. 1) GO TO 15                                      DO        71
      IDTYP=1                                                           DO        72
      CALL STORE                                                        DO        73
      LOC=NID                                                           DO        74
C** CHECK THAT IT IS AN INTEGER                                         DO        75
   15 CALL IMPTYP                                                       DO        76
      IF(BITGET(IDTBL(3,LOC),10,3) .NE. 4) CALL ERROR(40,NXTID)         DO        77
      IF(BITGET(IDTBL(3,LOC),1,1) .EQ. 1) CALL ERROR(14,NXTID)          DO        78
C** STORE DO PARAMETER IN BASIC BLOCK TABLE                             DO        79
      NBLOCK=NBLOCK+1                                                   DO        80
      IBLOCK(NBLOCK)=7000+LOC                                           DO        81
C** STORE LOOP IN SYMBOL TABLE                                          DO        82
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),ILOOP,36)                        DO        83
      PARAM(I)=0                                                        DO        84
   20 IF(I .EQ. 3) GO TO 30                                             DO        85
      IF(I .EQ. 1) GO TO 25                                             DO        86
      IF(NEXT(JPTR) .EQ. BLANK) GO TO 35                                DO        87
      JPTR=JPTR-1                                                       DO        88
   25 IF(NEXT(JPTR) .NE. COMMA) GO TO 50                                DO        89
   30 CONTINUE                                                          DO        90
      IF(NEXT(JPTR) .NE. BLANK) GO TO 50                                DO        91
C** CHECK SIZES OF DO PARAMETERS                                        DO        92
   35 IF(PARAM(1) .EQ. 0 .OR. PARAM(2) .EQ. 0) GO TO 40                 DO        93
      IF(PARAM(2) .LT. PARAM(1)) CALL ERROR(41)                         DO        94
      IF(PARAM(3) .EQ. 0) GO TO 40                                      DO        95
      IF((PARAM(2)+PARAM(3)-1) .GT. (2**17-2)) CALL ERROR(41)           DO        96
C** STORE BRANCH IN BASIC BLOCK TABLE                                   DO        97
   40 NBLOCK=NBLOCK+1                                                   DO        98
      IBLOCK(NBLOCK)=998                                                DO        99
      NBRNCH=1                                                          DO       100
      NB=1                                                              DO       101
      RETURN                                                            DO       102
   50 CALL ERROR(7)                                                     DO       103
      RETURN                                                            DO       104
      END                                                               DO       105
```

```
      SUBROUTINE EQUIV                                                  EQUIV     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH      2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGIO,NXTID,IDTYP,NID,LOC,        CY58A    80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH      4
      DIMENSION IALPH(11),IDIM(3),ISUB(3)                               EQUIV     4
      INTEGER BITPUT,BITGET                                             EQUIV     5
      INTEGER A,RPAR,COMMA,BOFFST,BLANK                                 EQUIV     6
      DATA (IALPH(I),I=1,11)/1HE,1HQ,1HU,1HI,1HV,1HA,1HL,1HE,1HN,1HC,  EQUIV     7
     1 1HE/                                                             EQUIV     8
      DATA LPAR/1H(/,RPAR/1H)/,COMMA/1H,/,BLANK/1H /                    EQUIV     9
C** EQUIVALENCE STATEMENT PROCESSOR                                     EQUIV    10
      DO 5 I=1,11                                                       EQUIV    11
      IF(NEXT(JPTR) .NE. IALPH(I)) GO TO 130                            EQUIV    12
    5 CONTINUE                                                          EQUIV    13
    8 IF(NEXT(JPTR) .NE. LPAR) GO TO 130                                EQUIV    14
      LSTLOC=0                                                          EQUIV    15
      BOFFST=0                                                          EQUIV    16
      J=0                                                               EQUIV    17
  120 J=J+1                                                             EQUIV    18
C** GET NEXT VARIABLE AND STORE IN SYMBOL TABLE                         EQUIV    19
      CALL GNLE                                                         EQUIV    20
      ILOC=1                                                            EQUIV    21
      IF(JTYP .NE. 2) GO TO 130                                         EQUIV    22
      CALL SEARCH                                                       EQUIV    23
      IF(ISRCH(2) .EQ. 1) CALL ERROR(10,NXTID)                          EQUIV    24
      IF(ISRCH(1) .EQ. 1) GO TO 9                                       EQUIV    25
      IDTYP=1                                                           EQUIV    26
      CALL STORE                                                        EQUIV    27
      LOC=NID                                                           EQUIV    28
    9 CALL IMPTYP                                                       EQUIV    29
      IF(BITGET(IDTBL(3,LOC),12,1) .EQ. 1) CALL ERROR(20,NXTID)         EQUIV    30
      IF(NEXT(JPTR) .NE. LPAR) GO TO 30                                 EQUIV    31
      IF(BITGET(IDTBL(3,LOC),1,1) .NE. 1) GO TO 150                     EQUIV    32
C** SUBSCRIPTED VARIABLE                                                EQUIV    33
      NDIM=BITGET(IDTBL(3,LOC),7,6)                                     EQUIV    34
      DO 10 I=1,NDIM                                                    EQUIV    35
C** GET NEXT SUBSCRIPT                                                  EQUIV    36
      CALL GNLE                                                         EQUIV    37
      IF(JTYP .NE. 5) GO TO 130                                         EQUIV    38
      IDIM(I)=N2                                                        EQUIV    39
      IF(N2 .LE. 0) CALL ERROR(8)                                       EQUIV    40
      IWRD=3+I/2                                                        EQUIV    41
      IPOS=18*MOD(I,2)+18                                               EQUIV    42
C** GET CORRESPONDING DIMENSION FROM SYMBOL TABLE                       EQUIV    43
      ISUB(I)=BITGET(IDTBL(IWRD,LOC),IPOS,18)                           EQUIV    44
      IF(N2 .GT. ISUB(I)) CALL ERROR(18)                                EQUIV    45
      IF(NEXT(JPTR) .EQ. COMMA) GO TO 10                                EQUIV    46
      IF(A(JPTR-1) .NE. RPAR) GO TO 130                                 EQUIV    47
      GO TO 15                                                          EQUIV    48
   10 CONTINUE                                                          EQUIV    49
      GO TO 140                                                         EQUIV    50
C** CALCULATE DISTANCE OF SUBSCRIPT FROM BEGINNING OF ARRAY             EQUIV    51
   15 NDIM=I                                                            EQUIV    52
      ILOC=IDIM(1)                                                      EQUIV    53
      IF(NDIM .EQ. 1) GO TO 25                                          EQUIV    54
      ILOC=ILOC+(IDIM(2)-1)*ISUB(1)                                     EQUIV    55
      IF(NDIM .EQ. 2) GO TO 25                                          EQUIV    56
```

```
      ILOC=ILOC+(IDIM(3)-1)*ISUB(1)*ISUB(2)                              EQUIV      57
   25 IT=BITGET(IDTBL(3,LOC),10,3)                                       EQUIV      58
C** ADJUST DISTANCE IF DOUBLE PREC. OR COMPLEX                           EQUIV      59
      IF(IT .EQ. 2 .OR. IT .EQ. 3) ILOC=2*ILOC                           EQUIV      60
C** CALCULATE OFFSET OF VARIABLE FROM BEGINNING OF EQUIVALENCE LINK      EQUIV      61
      IOFFST=1-ILOC-BOFFST                                               EQUIV      62
      GO TO 45                                                           EQUIV      63
   30 IF(BITGET(IDTBL(3,LOC),1,1) .NE. 0) CALL ERROR(14,NXTID)           EQUIV      64
C** NON-SUBSCRIPTED VARIABLE, SET OFFSET=BASE OFFSET                     EQUIV      65
      IOFFST=BOFFST                                                      EQUIV      66
      JPTR=JPTR-1                                                        EQUIV      67
   45 IF(BITGET(IDTBL(3,LOC),17,1) .EQ. 1) GO TO 57                      EQUIV      68
C** VARIABLE NOT PREVIOUSLY EQUIVALENCED,SET "EQUIVALENCED" FLAG         EQUIV      69
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),1,17)                             EQUIV      70
C** BRANCH IF NO VARIABLES YET IN EQUIVALENCE LINK                       EQUIV      71
      IF(LSTLOC .EQ. 0) GO TO 50                                         EQUIV      72
C** SET POINTER FROM PREVIOUS VARIABLE TO PRESENT VARIABLE               EQUIV      73
      IDTBL(7,LSTLOC)=LOC                                                EQUIV      74
      GO TO 55                                                           EQUIV      75
C** SET BEGINNING OF EQUIVALENCE LINK                                    EQUIV      76
   50 IFSTLC=LOC                                                         EQUIV      77
C** STORE OFFSET OF VARIABLE IN SYMBOL TABLE                             EQUIV      78
   55 IDTBL(8,LOC)=IOFFST                                                EQUIV      79
C** RESET "LAST VARIABLE IN LINK" TO PRESENT VARIABLE                    EQUIV      80
      LSTLOC=LOC                                                         EQUIV      81
C** IF FIRST VARIABLE IN LINK, PROCESSING DONE                           EQUIV      82
      IF(J .NE. 1) GO TO 100                                             EQUIV      83
      GO TO 98                                                           EQUIV      84
C** VARIABLE HAS BEEN PREVIOUSLY EQUIVALENCED                            EQUIV      85
   57 LOC3=LOC                                                           EQUIV      86
   58 LOC3=IDTBL(7,LOC3)                                                 EQUIV      87
      IF(LOC3 .EQ. 0) GO TO 59                                           EQUIV      88
      IF(LOC3 .EQ. LOC) GO TO 60                                         EQUIV      89
      GO TO 58                                                           EQUIV      90
   59 JLOC=ILOC+IDTBL(8,LOC)                                             EQUIV      91
      IF(JLOC .NE. IDIS) CALL ERROR(20,IDTBL(1,LOC))                     EQUIV      92
      GO TO 100                                                          EQUIV      93
C** RE-OPEN LINK AND SET FIRST LOCATION TO PRESENT VARIABLE              EQUIV      94
   60 IF(LSTLOC .NE. 0) GO TO 63                                         EQUIV      95
      IFSTLC=LOC                                                         EQUIV      96
      GO TO 65                                                           EQUIV      97
   63 IDTBL(7,LSTLOC)=LOC                                                EQUIV      98
   65 LOC2=LOC                                                           EQUIV      99
C** THIS LOOP FINDS THE END OF THE LINK                                  EQUIV     100
   70 NXTLOC=IDTBL(7,LOC2)                                               EQUIV     101
      IF(NXTLOC .EQ. LOC) GO TO 75                                       EQUIV     102
      LOC2=NXTLOC                                                        EQUIV     103
      GO TO 70                                                           EQUIV     104
C** LINK IS RE-OPENED HERE                                              EQUIV     105
   75 IDTBL(7,LOC2)=0                                                    EQUIV     106
C** RESET LAST VARIABLE IN LINK TO PRESENT VARIABLE                      EQUIV     107
      LSTLOC=LOC2                                                        EQUIV     108
C** CALCULATE OFFSET DIFFERENCE BETWEEN PRESENT AND OLD OFFSETS          EQUIV     109
      IOFFDF=IOFFST-IDTBL(8,LOC)                                         EQUIV     110
      IF(IOFFDF) 80,98,90                                                EQUIV     111
C** NEW OFFSET IS LESS THAN OLD OFFSET                                   EQUIV     112
C** ALL OFFSETS FROM HERE ON IN EQUIVALENCE LINK MUST BE CHANGED         EQUIV     113
```

```
   80 LOC2=LOC                                                          EQUIV    114
   85 IDTBL(8,LOC2)=IDTBL(8,LOC2)+IOFFDF                                EQUIV    115
      LOC2=IDTBL(7,LOC2)                                                EQUIV    116
      IF(LOC2 .EQ. 0) GO TO 98                                          EQUIV    117
      GO TO 85                                                          EQUIV    118
C** NEW OFFSET GREATER THAN OLD OFFSET                                  EQUIV    119
C** OFFSETS OF PREVIOUS VARIABLES IN LINK MUST BE CHANGED               EQUIV    120
   90 LOC2=IFSTLC                                                       EQUIV    121
   95 IF(LOC2 .EQ. LOC) GO TO 97                                        EQUIV    122
      IDTBL(8,LOC2)=IDTBL(8,LOC2)-IOFFDF                                EQUIV    123
      LOC2=IDTBL(7,LOC2)                                                EQUIV    124
      GO TO 95                                                          EQUIV    125
C** RESET BASE OFFSET                                                   EQUIV    126
   97 BOFFST=BOFFST-IOFFDF                                              EQUIV    127
   98 IDIS=ILOC+IDTBL(8,LOC)                                            EQUIV    128
  100 IF(NEXT(JPTR) .EQ. COMMA) GO TO 120                               EQUIV    129
      IF(A(JPTR-1) .NE. RPAR) GO TO 130                                 EQUIV    130
      IF(J .EQ. 1) GO TO 130                                            EQUIV    131
C** CLOSE EQUIVALENCE LINK                                              EQUIV    132
      IDTBL(7,LSTLOC)=IFSTLC                                            EQUIV    133
      JK=0                                                              EQUIV    134
      LOC3=IFSTLC                                                       EQUIV    135
C** TRAVERSE EQUIVALENCE LINK                                           EQUIV    136
      I=0                                                               EQUIV    137
  110 I=I+1                                                             EQUIV    138
      LOC3=IDTBL(7,LOC3)                                                EQUIV    139
      IF(BITGET(IDTBL(3,LOC3),16,1) .EQ. 0) GO TO 105                   EQUIV    140
C** VARIABLE IN COMMON, INCREMENT COUNTER                               EQUIV    141
      JK=JK+1                                                           EQUIV    142
C** IF TOO MANY VARIABLES IN COMMON, ISSUE DIAGNOSTIC                   EQUIV    143
      IF(JK .GT. 1) CALL ERROR(21,IDTBL(1,LOC3),IDTBL(1,LOC1))          EQUIV    144
      LOC1=LOC3                                                         EQUIV    145
      LOC=LOC3                                                          EQUIV    146
C** CHECK TO SEE IF COMMON WAS EXTENDED                                 EQUIV    147
      CALL COMEXT                                                       EQUIV    148
  105 IF(LOC3 .NE. IFSTLC) GO TO 110                                    EQUIV    149
      IF(NEXT(JPTR) .EQ. BLANK) RETURN                                  EQUIV    150
      IF(A(JPTR-1) .EQ. COMMA) GO TO 8                                  EQUIV    151
  130 CALL ERROR(7)                                                     EQUIV    152
      RETURN                                                            EQUIV    153
  140 CALL ERROR(19)                                                    EQUIV    154
      RETURN                                                            EQUIV    155
  150 CALL ERROR(13,NXTID)                                              EQUIV    156
      RETURN                                                            EQUIV    157
      END                                                               EQUIV    158
```

```
      SUBROUTINE ERROR(IERROR,INUM,INUM2)                                ERROR     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),   RICH      2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,         CY58A    80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                   RICH      4
      COMMON/FLOW/IFL                                                    ERROR     4
C** ERROR MESSAGE GENERATOR                                              ERROR     5
      WRITE(6,1)                                                         ERROR     6
    1 FORMAT(1X,100H**************************************************** ERROR     7
     1**********************************************)                   ERROR     8
      GO TO (5,15,25,35,45,55,65,75,85,95,105,115,125,135,145,155,165,   ERROR     9
     A175,185,195,205,215,225,235,245,255,265,275,285,295,305,315,325,   ERROR    10
     * 335,345,355,365,375,385,395,405,415,425,435,445,455,465,475,485,  ERROR    11
     * 495,505,515,525,535,545,555,565,575,585,595,605,615,625,635,645,  ERROR    12
     $ 655,665,675,685,695,705,715,725,735,745,755,765,775,785,795,805,  ERROR    13
     $ 815,825,835,845,855,865,875,885,895,905,915,925,935,945),IERROR   CY58A    55
    5 WRITE(6, 10)                                                       ERROR    15
   10 FORMAT(6X,26H THIS STATEMENT IS ILLEGAL)                           ERROR    16
      GO TO 1000                                                         ERROR    17
   15 WRITE(6, 20)                                                       ERROR    18
   20 FORMAT(6X,31H THIS STATEMENT IS OUT OF ORDER)                      ERROR    19
      GO TO 1000                                                         ERROR    20
   25 WRITE(6, 30)                                                       ERROR    21
   30 FORMAT(6X,39H VALUE OF INTEGER CONSTANT IS TOO LARGE)              ERROR    22
      GO TO 1000                                                         ERROR    23
   35 WRITE(6, 40)                                                       ERROR    24
   40 FORMAT(6X,28H TOO MANY CONTINUATION CARDS)                         ERROR    25
      GO TO 1000                                                         ERROR    26
   45 WRITE(6, 50)                                                       ERROR    27
   50 FORMAT(6X,30H HOLLERITH STRING IS TOO LARGE)                       ERROR    28
      GO TO 1000                                                         ERROR    29
   55 WRITE(6, 60)                                                       ERROR    30
   60 FORMAT(6X,26H VARIABLE NAME IS TOO LONG)                           ERROR    31
      GO TO 1000                                                         ERROR    32
   65 WRITE(6, 70)                                                       ERROR    33
   70 FORMAT(6X,31H SYNTAX ERROR IN THIS STATEMENT)                      ERROR    34
      IF(ITYP .LE. 18 .AND. IFL .GT. 0) IFL=-1                           ERROR    35
      GO TO 1000                                                         ERROR    36
   75 WRITE(6, 80)                                                       ERROR    37
   80 FORMAT(6X,46H ARRAY DIMENSION IS OUTSIDE OF ALLOWABLE RANGE)       ERROR    38
      GO TO 1000                                                         ERROR    39
   85 WRITE(6, 90)                                                       ERROR    40
   90 FORMAT(6X,45H ILLEGAL VARIABLE DIMENSION IN THIS STATEMENT)        ERROR    41
      GO TO 1000                                                         ERROR    42
   95 WRITE(6,100) INUM                                                  ERROR    43
  100 FORMAT(6X,33H THE FUNCTION OR SUBROUTINE NAME ,A6,18H IS USED ILLE ERROR    44
     *GALLY)                                                             ERROR    45
      GO TO 1000                                                         ERROR    46
  105 WRITE(6,110) INUM                                                  ERROR    47
  110 FORMAT(6X,14H THE VARIABLE ,A6,32H HAS BEEN PREVIOUSLY DIMENSIONED ERROR    48
     *)                                                                  ERROR    49
      GO TO 1000                                                         ERROR    50
  115 WRITE(6,120) INUM                                                  ERROR    51
  120 FORMAT(6X,14H THE VARIABLE ,A6,26H HAS BEEN PREVIOUSLY TYPED)      ERROR    52
      GO TO 1000                                                         ERROR    53
  125 WRITE(6,130) INUM                                                  ERROR    54
  130 FORMAT(6X,14H THE VARIABLE ,A6,38H IS ILLEGALLY FOLLOWED BY A LEFT ERROR    55
     * PAREN)                                                            ERROR    56
```

```
      GO TO 1000                                                        ERROR     57
  135 WRITE(6,140) INUM                                                 ERROR     58
  140 FORMAT(6X,26H THE DIMENSIONED VARIABLE ,A6,16H IS USED ILLEGALLY) ERROR     59
      GO TO 1000                                                        ERROR     60
  145 WRITE(6,150) INUM                                                 ERROR     61
  150 FORMAT(6X,18H STATEMENT NUMBER ,I5,15H IS NOT DEFINED)            ERROR     62
      IF(IFL .GT. 0) IFL=-1                                             ERROR     63
      GO TO 1000                                                        ERROR     64
  155 WRITE(6,160) INUM                                                 ERROR     65
  160 FORMAT(6X,18H STATEMENT NUMBER ,I5,18H IS NOT REFERENCED)         ERROR     66
      GO TO 1000                                                        ERROR     67
  165 WRITE(6,170) INUM                                                 ERROR     68
  170 FORMAT(6X,18H ILLEGAL VARIABLE ,A6,10H IN COMMON)                 ERROR     69
      GO TO 1000                                                        ERROR     70
  175 WRITE(6,180)                                                      ERROR     71
  180 FORMAT(6X,43H VALUE OF ARRAY SUBSCRIPT EXCEEDS DIMENSION)         ERROR     72
      GO TO 1000                                                        ERROR     73
  185 WRITE(6,190)                                                      ERROR     74
  190 FORMAT(6X,25H ERROR IN ARRAY SUBSCRIPT)                           ERROR     75
      GO TO 1000                                                        ERROR     76
  195 WRITE(6,200) INUM                                                 ERROR     77
  200 FORMAT(6X,18H ILLEGAL VARIABLE ,A6,16H IS EQUIVALENCED)           ERROR     78
      GO TO 1000                                                        ERROR     79
  205 WRITE(6,210) INUM,INUM2                                           ERROR     80
  210 FORMAT(6X,22H THE COMMON VARIABLES ,A6,5H AND ,A6,17H ARE EQUIVALE ERROR    81
     *NCED)                                                             ERROR     82
      GO TO 1000                                                        ERROR     83
  215 WRITE(6,220)                                                      ERROR     84
  220 FORMAT(6X,19H ILLEGAL I/O DEVICE)                                 ERROR     85
      GO TO 1000                                                        ERROR     86
  225 WRITE(6,230)                                                      ERROR     87
  230 FORMAT(6X,37H ILLEGAL CHARACTER IN THIS EXPRESSION)               ERROR     88
      GO TO 1000                                                        ERROR     89
  235 WRITE(6,240) INUM                                                 ERROR     90
  240 FORMAT(6X,25H ILLEGAL SUBROUTINE NAME ,A6)                        ERROR     91
      GO TO 1000                                                        ERROR     92
  245 WRITE(6,250)                                                      ERROR     93
  250 FORMAT(6X,50H SUBROUTINE TABLE OVERFLOW - PROCESSING TERMINATED)  ERROR     94
      GO TO 1000                                                        ERROR     95
  255 WRITE(6,260) INUM                                                 ERROR     96
  260 FORMAT(6X,77H INCORRECT NUMBER OF ARGUMENTS IN CALLING SEQUENCE OF ERROR    97
     $ FUNCTION OR SUBROUTINE ,A6)                                      ERROR     98
      GO TO 1000                                                        ERROR     99
  265 WRITE(6,270)                                                      ERROR    100
  270 FORMAT(6X,19H ILLEGAL ASSIGNMENT)                                 ERROR    101
      GO TO 1000                                                        ERROR    102
  275 WRITE(6,280) INUM                                                 ERROR    103
  280 FORMAT(6X,14H THE VARIABLE ,A6,42H APPEARS IN A DATA STATEMENT AND ERROR    104
     * IN COMMON)                                                       ERROR    105
      GO TO 1000                                                        ERROR    106
  285 WRITE(6,290) INUM                                                 ERROR    107
  290 FORMAT(6X,14H THE VARIABLE ,A6,44H HAS PREVIOUSLY APPEARED IN A DA ERROR    108
     *TA STATEMENT)                                                     ERROR    109
      GO TO 1000                                                        ERROR    110
  295 WRITE(6,300) INUM                                                 ERROR    111
  300 FORMAT(6X,22H THE FORMAL PARAMETER ,A6,31H APPEARS IN THIS DATA ST ERROR    112
     *ATEMENT)                                                          ERROR    113
```

```
      GO TO 1000                                                        ERROR     114
  305 WRITE(6,310)                                                      ERROR     115
  310 FORMAT(6X,24H LIST SIZES DO NOT MATCH)                            ERROR     116
      GO TO 1000                                                        ERROR     117
  315 WRITE(6,320)                                                      ERROR     118
  320 FORMAT(6X,24H ILLEGAL STATEMENT LABEL)                            ERROR     119
      IF(IFL .GT. 0) IFL=-1                                             ERROR     120
      GO TO 1000                                                        ERROR     121
  325 WRITE(6,330)                                                      ERROR     122
  330 FORMAT(6X,26H DUPLICATE STATEMENT LABEL)                          ERROR     123
      IF(IFL .GT. 0) IFL=-1                                             ERROR     124
      GO TO 1000                                                        ERROR     125
  335 WRITE(6,340)                                                      ERROR     126
  340 FORMAT(6X,34H THIS STATEMENT CAN NOT BE REACHED)                  ERROR     127
      GO TO 1000                                                        ERROR     128
  345 WRITE(6,350)                                                      ERROR     129
  350 FORMAT(6X,31H DO LOOPS ARE IMPROPERLY NESTED)                     ERROR     130
      GO TO 1000                                                        ERROR     131
  355 WRITE(6,360)                                                      ERROR     132
  360 FORMAT(6X,32H FORMAT STATEMENT IS NOT LABELED)                    ERROR     133
      GO TO 1000                                                        ERROR     134
  365 WRITE(6,370)                                                      ERROR     135
  370 FORMAT(6X,20H ILLEGAL DO TERMINAL)                                ERROR     136
      GO TO 1000                                                        ERROR     137
  375 WRITE(6,380)                                                      ERROR     138
  380 FORMAT(6X,37H LAST EXECUTABLE STATEMENT IS ILLEGAL)               ERROR     139
      IF(IFL .GT. 0) IFL=-1                                             ERROR     140
      GO TO 1000                                                        ERROR     141
  385 WRITE(6,390) INUM                                                 ERROR     142
  390 FORMAT(6X,24H THE VARIABLE REFERENCE ,A6,18H IS NOT AN INTEGER)   ERROR     143
      GO TO 1000                                                        ERROR     144
  395 WRITE(6,400) INUM                                                 ERROR     145
  400 FORMAT(6X,27H THE DO PARAMETER OR INDEX ,A6,18H IS NOT AN INTEGER)ERROR     146
      GO TO 1000                                                        ERROR     147
  405 WRITE(6,410)                                                      ERROR     148
  410 FORMAT(6X,52H VALUE OF DO PARAMETER IS OUTSIDE OF ALLOWABLE RANGE)ERROR     149
      GO TO 1000                                                        ERROR     150
  415 WRITE(6,420)                                                      ERROR     151
  420 FORMAT(6X,32H COMPLEX EXPRESSIONS ARE ILLEGAL)                    ERROR     152
      GO TO 1000                                                        ERROR     153
  425 WRITE(6,430)                                                      ERROR     154
  430 FORMAT(6X,24H ILLEGAL VARIABLE FORMAT)                            ERROR     155
      GO TO 1000                                                        ERROR     156
  435 WRITE(6,440)                                                      ERROR     157
  440 FORMAT(6X,39H THIS STATEMENT SHOULD HAVE AN I/O LIST)             ERROR     158
      GO TO 1000                                                        ERROR     159
  445 WRITE(6,450)                                                      ERROR     160
  450 FORMAT(6X,50H STATEMENT FOLLOWING LOGICAL EXPRESSION IS ILLEGAL)  ERROR     161
      GO TO 1000                                                        ERROR     162
  455 WRITE(6,460)                                                      ERROR     163
  460 FORMAT(6X,44H REAL NUMBER LIES OUTSIDE OF ALLOWABLE RANGE)        ERROR     164
      GO TO 1000                                                        ERROR     165
  465 WRITE(6,470)                                                      ERROR     166
  470 FORMAT(6X,42H THIS EQUIVALENCE STATEMENT EXTENDS COMMON)          ERROR     167
      GO TO 1000                                                        ERROR     168
  475 WRITE(6,480)                                                      ERROR     169
  480 FORMAT(6X,40H ILLEGAL VARIABLE IN COMMON BLOCK SESCOM)            ERROR     170
```

```
      GO TO 1000                                                        ERROR    171
  485 WRITE(6,490) INUM                                                 ERROR    172
  490 FORMAT(6X,12H SUBPROGRAM ,A6,19H HAS INCORRECT TYPE)              ERROR    173
      GO TO 1000                                                        ERROR    174
  495 WRITE(6,500) INUM                                                 ERROR    175
  500 FORMAT(6X,23H WARNING - ARGUMENT NO.,I3,34H MAY HAVE INCORRECT DIM ERROR   176
     *ENSIONALITY)                                                      ERROR    177
      GO TO 1000                                                        ERROR    178
  505 WRITE(6,510) INUM                                                 ERROR    179
  510 FORMAT(6X,13H ARGUMENT NO.,I3,19H HAS INCORRECT TYPE)             ERROR    180
      GO TO 1000                                                        ERROR    181
  515 WRITE(6,520)                                                      ERROR    182
  520 FORMAT(6X,49H WARNING - THIS MODULE IS NOT IN THE SESCOMP LIST)   ERROR    183
      GO TO 1000                                                        ERROR    184
  525 WRITE(6,530) INUM                                                 ERROR    185
  530 FORMAT(6X,14H THE VARIABLE ,A6,29H PREVIOUSLY APPEARS IN COMMON)  ERROR    186
      GO TO 1000                                                        ERROR    187
  535 WRITE(6,540) INUM                                                 ERROR    188
  540 FORMAT(6X,13H ARGUMENT NO.,I3,11H IS INVALID)                     ERROR    189
      GO TO 1000                                                        ERROR    190
  545 WRITE(6,550) INUM                                                 ERROR    191
  550 FORMAT(6X,13H ARGUMENT NO.,I3,29H IS DESIGNATED LOGICAL OUTPUT)   ERROR    192
      GO TO 1000                                                        ERROR    193
  555 WRITE(6,560) INUM                                                 ERROR    194
  560 FORMAT(6X,29H ILLEGAL COMMON BLOCK NAME - ,A6)                    ERROR    195
      GO TO 1000                                                        ERROR    196
  565 WRITE(6,570) INUM                                                 ERROR    197
  570 FORMAT(6X,41H WARNING - VARIABLE TYPE IN COMMON BLOCK ,A6,41H DOES ERROR   198
     $ NOT AGREE WITH INTERFACE DEFINITION)                             ERROR    199
      GO TO 1000                                                        ERROR    200
  575 WRITE(6,580) INUM                                                 ERROR    201
  580 FORMAT(6X,14H COMMON BLOCK ,A6,19H HAS INCORRECT SIZE)            ERROR    202
      GO TO 1000                                                        ERROR    203
  585 WRITE(6,590)                                                      ERROR    204
  590 FORMAT(6X,58H EXTERNAL REFERENCE TABLE OVERFLOW - PROCESSING TERMI ERROR   205
     $NATED)                                                            ERROR    206
      GO TO 1000                                                        ERROR    207
  595 WRITE(6,600)                                                      ERROR    208
  600 FORMAT(6X,52H COMMON BLOCK TABLE OVERFLOW - PROCESSING TERMINATED) ERROR    209
      GO TO 1000                                                        ERROR    210
  605 WRITE(6,610) INUM                                                 ERROR    211
  610 FORMAT(6X,29H ILLEGAL COMMON BLOCK NAME - ,A6)                    ERROR    212
      GO TO 1000                                                        ERROR    213
  615 WRITE(6,620) INUM                                                 ERROR    214
  620 FORMAT(6X,14H COMMON BLOCK ,A6,27H IS NOT IN THE SESCOMP LIST)    ERROR    215
      GO TO 1000                                                        ERROR    216
  625 WRITE(6,630) INUM                                                 ERROR    217
  630 FORMAT(6X,25H CATEGORY 2 COMMON BLOCK ,A6,23H IS NOT GROUPED BY TY ERROR   218
     $PE)                                                               ERROR    219
      GO TO 1000                                                        ERROR    220
  635 WRITE(6,640) INUM,INUM2                                           ERROR    221
  640 FORMAT(6X,38H DOUBLE PRECISION OR COMPLEX VARIABLE ,A6,56H DOES NO ERROR   222
     $T BEGIN ON AN EVEN LOCATION WITHIN COMMON BLOCK ,A6)              ERROR    223
      GO TO 1000                                                        ERROR    224
  645 WRITE(6,650) INUM                                                 ERROR    225
  650 FORMAT(6X,26H VARIABLE IN COMMON BLOCK ,A6,16H IS OUT OF ORDER)   ERROR    226
      GO TO 1000                                                        ERROR    227
```

```
655 WRITE(6,660)                                                        ERROR    228
660 FORMAT(6X,56H THE COMMON BLOCK SESCOM DOES NOT APPEAR IN THIS PROG ERROR    229
   $RAM)                                                                ERROR    230
    GO TO 1000                                                          ERROR    231
665 WRITE(6,670) INUM                                                   ERROR    232
670 FORMAT(6X,14H THE DO INDEX ,A6,13H IS REDEFINED)                    ERROR    233
    RETURN                                                              ERROR    234
675 WRITE(6,680) INUM                                                   ERROR    235
680 FORMAT(6X,24H THE VARIABLE DIMENSION ,A6,13H IS REDEFINED)          ERROR    236
    RETURN                                                              ERROR    237
685 WRITE(6,690) INUM                                                   ERROR    238
690 FORMAT(6X,23H THE ASSIGNED VARIABLE ,A6,24H IS ILLEGALLY REFERENCE ERROR    239
   $D)                                                                  ERROR    240
    RETURN                                                              ERROR    241
695 WRITE(6,700) INUM                                                   ERROR    242
700 FORMAT(6X,14H THE VARIABLE ,A6,30H IS REFERENCED BUT NOT DEFINED)   ERROR    243
    RETURN                                                              ERROR    244
705 WRITE(6,710) INUM                                                   ERROR    245
710 FORMAT(6X,14H THE VARIABLE ,A6,45H IS REFERENCED ILLEGALLY BY AN A ERROR    246
   $SSIGNED GO TO)                                                      ERROR    247
    RETURN                                                              ERROR    248
715 WRITE(6,720) INUM                                                   ERROR    249
720 FORMAT(6X,18H THE DO PARAMETER ,A6,13H IS REDEFINED)                ERROR    250
    RETURN                                                              ERROR    251
725 WRITE(6,730)                                                        ERROR    252
730 FORMAT(6X,49H THIS MODULE CONTAINS NO CATEGORY 2 COMMON BLOCKS)     ERROR    253
    GO TO 1000                                                          ERROR    254
735 WRITE(6,740) INUM                                                   ERROR    255
740 FORMAT(6X,24H THE ANSI FUNCTION NAME ,A6,27H IS MISUSED IN THIS PR ERROR    256
   $OGRAM)                                                              ERROR    257
    GO TO 1000                                                          ERROR    258
745 WRITE(6,750) INUM                                                   ERROR    259
750 FORMAT(6X,14H THE VARIABLE ,A6,60H APPEARS IN A CATEGORY 2 OR 3 CO ERROR    260
   $MMON BLOCK BUT IS NEVER USED)                                       ERROR    261
    GO TO 1000                                                          ERROR    262
755 WRITE(6,760)                                                        ERROR    263
760 FORMAT(6X,75H ARRAY SUBSCRIPT OR IMPLIED DO PARAMETER MAY LIE OUTS ERROR    264
   $IDE OF ALLOWABLE RANGE)                                             ERROR    265
    GO TO 1000                                                          ERROR    266
765 WRITE(6,770) INUM,INUM2                                             ERROR    267
770 FORMAT(6X,22H MIXED MODE COMBINING ,A6,6H WITH ,A6)                 ERROR    268
    GO TO 1000                                                          ERROR    269
775 WRITE(6,780) INUM                                                   ERROR    270
780 FORMAT(6X,33H INCORRECT EXPONENT AT CHAR. NO. ,I3)                  ERROR    271
    GO TO 1000                                                          ERROR    272
785 WRITE(6,790) INUM                                                   ERROR    273
790 FORMAT(6X,47H VAR-CONST CONFUSION IN SUBSCRIPT AT CHAR. NO. ,I3)    ERROR    274
    GO TO 1000                                                          ERROR    275
795 WRITE(6,800) INUM,INUM2                                             ERROR    276
800 FORMAT(6X,40H SUBSCRIPT CONSTANT OR VARIABLE OF TYPE ,A6,14H AT CH ERROR    277
   $AR. NO. ,I3)                                                        ERROR    278
    GO TO 1000                                                          ERROR    279
805 WRITE(6,810) INUM                                                   ERROR    280
810 FORMAT(6X,52H TOO MANY SUBSCRIPTS FOR THIS VARIABLE AT CHAR. NO. , ERROR    281
   $I3)                                                                 ERROR    282
    GO TO 1000                                                          ERROR    283
815 WRITE(6,820) INUM                                                   ERROR    284
```

```
820 FORMAT(6X,51H TOO FEW SUBSCRIPTS FOR THIS VARIABLE AT CHAR. NO. ,  ERROR  285
   $I3)                                                                ERROR  286
    GO TO 1000                                                         ERROR  287
825 WRITE(6,830) INUM                                                  ERROR  288
830 FORMAT(6X,52H ILLEGAL TYPE IN RELATIONAL EXPRESSION AT CHAR. NO. ,  ERROR  289
   $I3)                                                                ERROR  290
    GO TO 1000                                                         ERROR  291
835 WRITE(6,840) INUM                                                  ERROR  292
840 FORMAT(6X,40H TOO MANY ARGUMENTS IN CALLING SEQUENCE ,I3)          ERROR  293
    GO TO 1000                                                         ERROR  294
845 WRITE(6,850)                                                       ERROR  295
850 FORMAT(6X,41H TOO MANY FUNCTION REFS IN THIS STATEMENT)            ERROR  296
    GO TO 1000                                                         ERROR  297
855 WRITE(6,860) INUM                                                  ERROR  298
860 FORMAT(6X,26H INVALID FORMAL PARAMETER ,A6)                        ERROR  299
    GO TO 1000                                                         ERROR  300
865 WRITE(6,870) INUM                                                  ERROR  301
870 FORMAT(6X,19H THE FUNCTION NAME ,A6,33H MAY HAVE BEEN PREVIOUSLY M ERROR  302
   *ISUSED)                                                            ERROR  303
    GO TO 1000                                                         ERROR  304
875 WRITE(6,880) INUM                                                  ERROR  305
880 FORMAT(6X,39H ILLEGAL FIELD DESCRIPTOR AT CHAR. NO. ,I4)           ERROR  306
    GO TO 1000                                                         CY58A   56
885 WRITE(6,890)                                                       CY58A   57
890 FORMAT(6X,38H TOO MANY FUNCTION DEFINING STATEMENTS)               CY58A   58
    GO TO 1000                                                         CY58A   59
895 WRITE(6,900)                                                       CY58A   60
900 FORMAT(6X,71H TOO MANY EXTERNAL REFERENCES IN THIS STATEMENT - PRO CY58A   61
   $CESSING TERMINATED)                                                CY58A   62
    GO TO 1000                                                         CY58A   63
905 WRITE(6,910)                                                       CY58A   64
910 FORMAT(6X,46H STATEMENT IS TOO LONG - PROCESSING TERMINATED)       CY58A   65
    GO TO 1000                                                         CY58A   66
915 WRITE(6,920)                                                       CY58A   67
920 FORMAT(6X,46H SESCOMP LIST OVERFLOW - PROCESSING TERMINATED)       CY58A   68
    GO TO 1000                                                         CY58A   69
925 WRITE(6,930)                                                       CY58A   70
930 FORMAT(6X,63H OVERFLOW OF INTERFACE DEFINITION TABLE - PROCESSING   CY58A   71
   $TERMINATED)                                                        CY58A   72
    GO TO 1000                                                         CY58A   73
935 WRITE(6,940)                                                       CY58A   74
940 FORMAT(6X,32H TOO MANY EQUIVALENCED VARIABLES)                     CY58A   75
    GO TO 1000                                                         CY58A   76
945 WRITE(6,950)                                                       CY58A   77
950 FORMAT(6X,61H TOO MANY VARIABLES IN THIS STATEMENT - PROCESSING TE CY58A   78
   $RMINATED)                                                          CY58A   79
1000 WRITE(6,1)                                                        ERROR  307
    RETURN                                                             ERROR  308
    END                                                                ERROR  309
```

```
      SUBROUTINE EXPR                                                    EXPR     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),   RICH     2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,         CY58A   80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                   RICH     4
      COMMON/FUNC/IFNCRA(5,12),MARGS,IARGS(50),FNCLOC(5),NFUNC           CY58A   13
      COMMON/STRING/NTYPE,NSTR,STR(500)                                  EXPR     5
      COMMON/LIST/NLIST,NINTFC,ISUBLT(2,200),INTFAC(300)                 EXPR     6
      COMMON/BASBLK/IBLOCK(2500),NBLOCK,NB,NBRNCH                        CY58A   14
      INTEGER FNCLOC,OPRA(6),BITPUT,BITGET                               EXPR     8
      INTEGER D,ASTRIK,DEE,EQUALS,STR,A,COMMA,RPAR,BLANK                 EXPR     9
      DATA (OPRA(I),I=1,6)/1H+,1H-,1H/,1H(,1H),1H,/,ASTRIK/1H*/,DEE/1HD/ EXPR    10
     1 ,EQUALS/1H=/,COMMA/1H,/,RPAR/1H)/,LPAR/1H(/,BLANK/1H /            EXPR    11
C** EXPRESSION PROCESSOR                                                 EXPR    12
C** THIS ROUTINE ENCODES ARITHMETIC AND LOGICAL EXPRESSIONS              EXPR    13
C** AND I/O LISTS FOR INPUT TO THE PARSER                                EXPR    14
      LP=0                                                               EXPR    15
      NFUNC=0                                                            EXPR    16
      K=0                                                                EXPR    17
      IEXPST=NBLOCK+1                                                    EXPR    18
      MARGS=0                                                            EXPR    19
  200 K=K+1                                                              EXPR    20
C** GET NEXT LANGUAGE ELEMENT IN STATEMENT                               EXPR    21
      CALL GNLE                                                          EXPR    22
C** NO MORE CHARACTERS LEFT, RETURN                                      EXPR    23
      IF(JTYP .EQ. 0) RETURN                                             EXPR    24
C** NOT A SPECIAL CHARACTER, KEEP GOING                                  EXPR    25
      IF(JTYP .NE. 1) GO TO 20                                           EXPR    26
      IF(LTYP .EQ. 9 .OR. ITYP .EQ. 6) GO TO 2                           EXPR    27
      IF(ITYP .EQ. 1 .OR. ITYP .EQ. 35) GO TO 1                          EXPR    28
      GO TO 5                                                            EXPR    29
C** CHECK FOR END OF EXPRESSION IN "IF" STATEMENT                        EXPR    30
C** EQUAL SIGN TERMINATES STRING                                         EXPR    31
    1 IF(D(1) .EQ. EQUALS) RETURN                                        EXPR    32
      GO TO 5                                                            EXPR    33
    2 IF(D(1) .EQ. RPAR .AND. LP .EQ. 1) RETURN                          EXPR    34
C** SPECIAL CHARACTER LOOP                                               EXPR    35
    5 DO 10 I=1,6                                                        EXPR    36
      IF(D(1) .NE. OPRA(I)) GO TO 10                                     EXPR    37
C** ENCODE SPECIAL CHARACTER                                             EXPR    38
      STR(K)=-I                                                          EXPR    39
      IF(I .EQ. 4) GO TO 6                                               EXPR    40
      IF(I .EQ. 5) GO TO 7                                               EXPR    41
      GO TO 100                                                          EXPR    42
C** LEFT PAREN FOUND - INCREMENT COUNTER                                 EXPR    43
    6 LP=LP+1                                                            EXPR    44
      GO TO 100                                                          EXPR    45
C** RIGHT PAREN FOUND - DECREMENT COUNTER                                EXPR    46
    7 LP=LP-1                                                            EXPR    47
      GO TO 100                                                          EXPR    48
   10 CONTINUE                                                           EXPR    49
      IF(D(1) .NE. EQUALS) GO TO 12                                      EXPR    50
C** ENCODE EQUALS SIGN                                                   EXPR    51
      STR(K)=-18                                                         EXPR    52
      GO TO 100                                                          EXPR    53
   12 IF(D(1) .NE. ASTRIK) GO TO 110                                     EXPR    54
      IF(D(2) .EQ. ASTRIK .AND. M .GT. 1) GO TO 15                       EXPR    55
C** ENCODE ASTRIK                                                        EXPR    56
```

```
      STR(K)=-7                                                    EXPR    57
      GO TO 100                                                    EXPR    58
C** ENCODE EXPONENTIATION SIGN                                     EXPR    59
   15 STR(K)=-8                                                    EXPR    60
      GO TO 100                                                    EXPR    61
   20 IF(JTYP .NE. 7) GO TO 30                                     EXPR    62
      IF(LOGID .GT. 9) GO TO 25                                    EXPR    63
C** ENCODE LOGICAL OPERATOR                                        EXPR    64
      STR(K)=-(LOGID+8)                                            EXPR    65
      GO TO 100                                                    EXPR    66
C** ENCODE LOGICAL CONSTANT                                        EXPR    67
   25 STR(K)=LSTART+440000+M*1000000                               EXPR    68
      GO TO 100                                                    EXPR    69
   30 IF(JTYP .NE. 4) GO TO 40                                     EXPR    70
      IF(IDES .EQ. 0) GO TO 35                                     EXPR    71
C** ENCODE DOUBLE PRECISION CONSTANT                               EXPR    72
      STR(K)=LSTART+420000+M*1000000                               EXPR    73
      GO TO 100                                                    EXPR    74
   35 CONTINUE                                                     EXPR    75
C** ENCODE REAL CONSTANT                                           EXPR    76
      STR(K)=LSTART+400000+M*1000000                               EXPR    77
      GO TO 100                                                    EXPR    78
   40 IF(JTYP .NE. 6) GO TO 50                                     EXPR    79
C** ENCODE COMPLEX CONSTANT                                        EXPR    80
      STR(K)=LSTART+410000+M*1000000                               EXPR    81
      GO TO 100                                                    EXPR    82
   50 IF(JTYP .NE. 5) GO TO 55                                     CY58A   15
C** ENCODE INTEGER                                                 EXPR    84
      STR(K)=LSTART+430000+M*1000000                               EXPR    85
      GO TO 100                                                    EXPR    86
   55 IF(JTYP .NE. 3) GO TO 60                                     CY58A   16
      STR(K)=LSTART+450000+M*1000000                               CY58A   17
      IF(ITYP .NE. 8) CALL ERROR(23)                               CY58A   18
      GO TO 100                                                    CY58A   19
   60 IF(JTYP .NE. 2) GO TO 110                                    EXPR    87
C** VARIABLE FOUND - SEARCH SYMBOL TABLE                           EXPR    88
      CALL SEARCH                                                  EXPR    89
      IBETA=0                                                      EXPR    90
      IF(NEXT(JPTR) .NE. LPAR) GO TO 64                            EXPR    91
      IF(ISRCH(1) .EQ. 0) GO TO 62                                 EXPR    92
      IF(BITGET(IDTBL(3,LOC),1,1) .EQ. 1) GO TO 67                 EXPR    93
C** VARIABLE IS NOT DIMENSIONED - MUST BE A FUNCTION               EXPR    94
C** CHANGE STORAGE IN SYMBOL TABLE                                 EXPR    95
      CALL SWITCH                                                  EXPR    96
      IBETA=5                                                      EXPR    97
      GO TO 63                                                     EXPR    98
C** FUNCTION REFERENCE                                             EXPR    99
   62 IBETA=5                                                      EXPR   100
      IF(ISRCH(2) .EQ. 1) GO TO 63                                 EXPR   101
C** FUNCTION NOT YET STORED                                        EXPR   102
      IDTYP=2                                                      EXPR   103
      CALL STORE                                                   EXPR   104
      LOC=NID                                                      EXPR   105
      DO 70 I=1,NLIST                                              EXPR   106
      IF(ISUBLT(1,I) .NE. IDTBL(1,LOC)) GO TO 70                   EXPR   107
C** FUNCTION NAME FOUND IN SESCOMP LIST                            EXPR   108
      IF(BITGET(ISUBLT(2,I),10,4) .NE. 4) GO TO 63                 EXPR   109
```

```
C** INTRINSIC FUNCTION - STORE TYPE IN SYMBOL TABLE                    EXPR    110
      ITP=BITGET(ISUBLT(2,I),13,3)                                     EXPR    111
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),ITP,10)                         EXPR    112
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),1,11)                           EXPR    113
      GO TO 63                                                         EXPR    114
   70 CONTINUE                                                         EXPR    115
C** PUT FUNCTION IN FUNCTION LIST                                      EXPR    116
   63 NFUNC=NFUNC+1                                                    EXPR    117
      IF(NFUNC .GT. 5) GO TO 120                                       CY58A    20
      FNCLOC(NFUNC)=LOC                                                EXPR    118
      GO TO 68                                                         EXPR    119
C** NON-DIMENSIONED VARIABLE                                          EXPR    120
   64 IF(ISRCH(2) .NE. 1) GO TO 65                                     EXPR    121
C** FUNCTION NAME NOT FOLLOWED BY LEFT PAREN, MUST BE THIS FUNCTION    EXPR    122
      IF(NXTID .NE. IFNCNM) CALL ERROR(10,NXTID)                       EXPR    123
C** STORE IN SYMBOL TABLE                                             EXPR    124
   65 IF(ISRCH(1) .EQ. 1) GO TO 67                                     EXPR    125
      IDTYP=1                                                          EXPR    126
      CALL STORE                                                       EXPR    127
      LOC=NID                                                          EXPR    128
      GO TO 68                                                         EXPR    129
C** SET DIMENSIONALITY                                                EXPR    130
   67 IBETA=BITGET(IDTBL(3,LOC),7,6)                                   EXPR    131
      IF(NXTID .NE. IFNCNM) GO TO 68                                   EXPR    132
      IBETA=0                                                          EXPR    133
      LOC=IDES                                                         EXPR    134
   68 CALL IMPTYP                                                      EXPR    135
C** SET TYPE                                                          EXPR    136
      IALPH=BITGET(IDTBL(3,LOC),10,3)-1                                EXPR    137
      JPTR=JPTR-1                                                      EXPR    138
C** ENCODE VARIABLE                                                   EXPR    139
      STR(K)=LOC+10000*IALPH+100000*IBETA+1000000*M                    EXPR    140
  100 NSTR=K                                                           EXPR    141
      IF(NSTR .GT. 500) GO TO 130                                      CY58A    21
      GO TO 200                                                        EXPR    142
  110 CALL ERROR(23)                                                   EXPR    143
      RETURN                                                           EXPR    144
  120 CALL ERROR(90)                                                   CY58A    22
      STOP                                                             CY58A    23
  130 CALL ERROR(91)                                                   CY58A    24
      STOP                                                             CY58A    25
      END                                                              EXPR    145
```

```
      SUBROUTINE EXPRCK                                                EXPRCK    2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH     2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A   80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                 RICH     4
      COMMON/TYP/NQQ,RHSTYP,NQ2,NQ3,LHSTYP                             EXPRCK    4
      DIMENSION IA(5,5)                                                EXPRCK    5
      INTEGER RHSTYP                                                   EXPRCK    6
      DATA ((IA(I,J),I=1,5),J=1,5)/1,0,0,1,0,0,1,0,0,0,1,0,1,1,0,      EXPRCK    7
     1   1,0,1,1,0,0,0,0,0,1/                                          EXPRCK    8
C** THIS ROUTINE IS CALLED BY THE ASSIGNMENT STATEMENT PROCESSOR TO    EXPRCK    9
C** CHECK THE "LEFT SIDE TYPE" AND "RIGHT SIDE TYPE" TO SEE IF THE     EXPRCK   10
C** ASSIGNMENT IS VALID                                               EXPRCK   11
      IF(IA(LHSTYP,RHSTYP+1) .EQ. 0) CALL ERROR(27)                    EXPRCK   12
      RETURN                                                           EXPRCK   13
      END                                                              EXPRCK   14
```

```
      SUBROUTINE FLOWCK                                                  FLOWCK     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),   RICH     2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A     80
     2 LTYP,ITYP,IBLKOT,MODE,IERR,IOES                                   RICH      4
      COMMON/BASBLK/IBLOCK(2500),NBLOCK,NB,NBRNCH                        CY58A     50
      COMMON/DOLOOP/ISTACK(4,50),NSTACK,ILOOP,IOVFLW                     FLOWCK    5
      COMMON/LABELS/STATRA(2,200),NLABEL                                 FLOWCK    6
      COMMON/FLOW/IFL,IRP                                                FLOWCK    7
      DIMENSION IPATH(100),ISTCK(100)                                    FLOWCK    8
      INTEGER FLWLST(100),BRANCH,STATRA                                  FLOWCK    9
      INTEGER BITPUT,BITGET                                              FLOWCK    10
      EQUIVALENCE (IPATH(1),A(1)),(ISTCK(1),A(101)),(FLWLST(1),A(201))   FLOWCK    11
C** FLOW ANALYSIS ALGORITHM - CHECKS EVERY POSSIBLE PATH OF FLOW         FLOWCK    12
C** THROUGH THE PROGRAM                                                 FLOWCK    13
      IF(IFL .EQ. -1) GO TO 3000                                         FLOWCK    14
      CLTM1=SECOND(T)                                                    FLOWCK    15
      IRX=0                                                              FLOWCK    16
      NSTCK=0                                                            FLOWCK    17
      NFLOW=0                                                            FLOWCK    18
      NOC=0                                                              FLOWCK    19
      NPTHS=0                                                            FLOWCK    20
C** START WITH FIRST BASIC BLOCK                                         FLOWCK    21
      IBLKST=1                                                           FLOWCK    22
C** SET INITIALLY DEFINED VARIABLES                                      FLOWCK    23
      CALL CHKLST                                                        FLOWCK    24
      WRITE(6,8)                                                         FLOWCK    25
    8 FORMAT(1H1,38H****** RESULTS OF FLOW ANALYSIS ******//)            FLOWCK    26
    5 DO 10 I=1,NID                                                      FLOWCK    27
   10 IDTBL(2,I)=IDTBL(8,I)                                              FLOWCK    28
   12 IF(NFLOW .EQ. 0) GO TO 20                                          FLOWCK    29
      NOC=0                                                              FLOWCK    30
C** THIS COMPUTES THE NUMBER OF DUPLICATE OCCURANCES FOR THE             FLOWCK    31
C** CURRENT BASIC BLOCK                                                 FLOWCK    32
      DO 15 I=1,NFLOW                                                    FLOWCK    33
      IF(IABS(FLWLST(I)) .NE. IBLKST) GO TO 15                           FLOWCK    34
      NOC=NOC+1                                                          FLOWCK    35
   15 CONTINUE                                                           FLOWCK    36
C** TERMINATE FLOW ANALYSIS FOR THIS PATH IF TOO MANY OCCURANCES         FLOWCK    37
      IF(NOC .GT. IRP) GO TO 1500                                        FLOWCK    38
C** ADD BLOCK TO CURRENT FLOW PATH                                       FLOWCK    39
   20 NFLOW=NFLOW+1                                                      FLOWCK    40
      IF(NFLOW .GT. 100) GO TO 4000                                      FLOWCK    41
      FLWLST(NFLOW)=IBLKST                                               FLOWCK    42
C** GET END OF BLOCK                                                     FLOWCK    43
      IEND=BITGET(IBLOCK(IBLKST),28,16)-1                                FLOWCK    44
      IF(IEND .EQ. -1) IEND=NBLOCK                                       FLOWCK    45
C** GET NUMBER OF BRANCHES FROM BLOCK                                    FLOWCK    46
      NBR=BITGET(IBLOCK(IBLKST),6,6)                                     FLOWCK    47
C** GET BLOCK OF NEXT BRANCH                                             FLOWCK    48
      ISTART=IEND-NBR+1                                                  FLOWCK    49
      IBLKST=NXTBLK(ISTART,IEND)                                         FLOWCK    50
      IF(NBR .EQ. 1) GO TO 25                                            FLOWCK    51
      FLWLST(NFLOW)=-FLWLST(NFLOW)                                       FLOWCK    52
C** MORE THAN ONE BRANCH                                                 FLOWCK    53
C** STORE NEXT BLOCK ON STACK AS NEGATIVE NUMBER                         FLOWCK    54
      NSTCK=NSTCK+1                                                      FLOWCK    55
      IF(NSTCK .GT. 100) GO TO 5000                                      FLOWCK    56
```

```
      ISTCK(NSTCK)=-NXTBLK(IEND,IEND)                                   FLOWCK     57
      IF(NBR .EQ. 2) GO TO 25                                           FLOWCK     58
C** MORE THAN TWO BRANCHES - PUT ALL REMAINING BRANCHES ON STACK        FLOWCK     59
      DO 22 J=3,NBR                                                     FLOWCK     60
      NSTCK=NSTCK+1                                                     FLOWCK     61
      IF(NSTCK .GT. 100) GO TO 5000                                     FLOWCK     62
   22 ISTCK(NSTCK)=NXTBLK(IEND-J+2,IEND)                                FLOWCK     63
   25 IF(IBLKST .NE. 0) GO TO 12                                        FLOWCK     64
C** ALL BRANCHES HAVE BEEN PLACED ON CURRENT FLOW PATH OR THE STACK      FLOWCK     65
C** BEGIN TRACING THROUGH PATH                                          FLOWCK     66
      NPATH=0                                                           FLOWCK     67
C** INCREMENT PATH COUNTER                                             FLOWCK     68
      NPTHS=NPTHS+1                                                     FLOWCK     69
      DO 1000 I=1,NFLOW                                                 FLOWCK     70
C** GET NEXT BLOCK IN PATH                                             FLOWCK     71
      BRANCH=IABS(FLWLST(I))                                            FLOWCK     72
C** SET POINTER TO FIRST VARIABLE IN BLOCK                             FLOWCK     73
      ISTART=BRANCH+1                                                   FLOWCK     74
C** GET START OF NEXT BLOCK                                            FLOWCK     75
      NXBLOK=BITGET(IBLOCK(BRANCH),28,16)                               FLOWCK     76
      IF(NXBLOK .EQ. 0) NXBLOK=NBLOCK+1                                 FLOWCK     77
C** GET STATEMENT NUMBER OF BLOCK                                      FLOWCK     78
      ISL=BITGET(IBLOCK(BRANCH),36,8)                                   FLOWCK     79
C** GET DO LOOP CONTAINING BLOCK                                       FLOWCK     80
      ILOOP=BITGET(IBLOCK(BRANCH),12,6)                                 FLOWCK     81
C** GET NUMBER OF BRANCHES FROM BLOCK                                  FLOWCK     82
      NBR=BITGET(IBLOCK(BRANCH),6,6)                                    FLOWCK     83
C** SET POINTER TO LAST VARIABLE IN BLOCK                              FLOWCK     84
      IEND=NXBLOK-NBR-1                                                 FLOWCK     85
C** BLOCK HAS STATEMENT LABEL - STORE IN STATEMENT LABEL LIST           FLOWCK     86
C** SET "USED" FLAG                                                    FLOWCK     87
      IF(ISL .EQ. 0) GO TO 45                                           FLOWCK     88
      NPATH=NPATH+1                                                     FLOWCK     89
      IPATH(NPATH)=STATRA(1,ISL)                                        FLOWCK     90
      STATRA(2,ISL)=BITPUT(STATRA(2,ISL),1,18)                          FLOWCK     91
   45 IF(IBLOCK(ISTART) .LT. 1000) GO TO 1000                           FLOWCK     92
C** THIS LOOP EXAMINES ALL VARIABLES IN THE BLOCK                      FLOWCK     93
      DO 500 J=ISTART,IEND                                              FLOWCK     94
C** GET VARIABLE CLASS          1 - DEFINED   2 - REFERENCED   3 - DO INDEX FLOWCK  95
C**                             4 - ASSIGNED  5 - REFERENCED BY ASSN. GOTO FLOWCK  96
C**                             6 - MADE UNDEFINED   7 - DO PARAMETER     FLOWCK     97
      IT=IBLOCK(J)/1000                                                 FLOWCK     98
C** GET SYMBOL TABLE LOCATION OF VARIABLE                              FLOWCK     99
      LOC=IBLOCK(J)-IT*1000                                             FLOWCK    100
      GO TO(50,60,70,80,90,100,200),IT                                  FLOWCK    101
C** VARIABLE IS DEFINED                                                FLOWCK    102
   50 IF(BITGET(IDTBL(3,LOC),13,1) .EQ. 1) GO TO 120                    FLOWCK    103
      IF(IDTBL(2,LOC) .EQ. 2) GO TO 55                                  FLOWCK    104
      IF(IDTBL(2,LOC) .EQ. 4) GO TO 180                                 FLOWCK    105
C** SET DEFINED FLAG                                                   FLOWCK    106
   52 IDTBL(2,LOC)=1                                                    FLOWCK    107
   30 IF(BITGET(IDTBL(3,LOC),17,1) .EQ. 0) GO TO 500                    FLOWCK    108
C** VARIABLE IS EQUIVALENCED - GET TYPE                                FLOWCK    109
      KTYPE=BITGET(IDTBL(3,LOC),10,3)                                   FLOWCK    110
C** SET ALL VARIABLES OF SAME TYPE WHICH ARE EQUIVALENCED TO THIS ONE   FLOWCK    111
C** TO "DEFINED"                                                       FLOWCK    112
C** SET ALL VARIABLES OF DIFFERENT TYPE TO "UNDEFINED"                  FLOWCK    113
```

```
      NXQV=LOC                                                          FLOWCK    114
   53 NXQV=IDTBL(7,NXQV)                                                FLOWCK    115
      IF(NXQV .EQ. LOC) GO TO 500                                       FLOWCK    116
      IF(BITGET(IDTBL(3,NXQV),10,3) .EQ. KTYPE) GO TO 54                FLOWCK    117
      IDTBL(2,NXQV)=0                                                   FLOWCK    118
      GO TO 53                                                          FLOWCK    119
   54 IDTBL(2,NXQV)=1                                                   FLOWCK    120
      GO TO 53                                                          FLOWCK    121
C** FLAG INDICATES THAT VARIABLE WAS ONCE A DO INDEX                    FLOWCK    122
C** MAKE SURE THAT IT IS NOT CURRENTLY A DO INDEX                       FLOWCK    123
   55 IF(ILOOP .EQ. 0) GO TO 52                                         FLOWCK    124
   57 IF(LOC .EQ. ISTACK(4,ILOOP)) GO TO 110                            FLOWCK    125
      IF(ISTACK(3,ILOOP) .EQ. 0) GO TO 52                               FLOWCK    126
      ILOOP=ISTACK(3,ILOOP)                                             FLOWCK    127
      GO TO 57                                                          FLOWCK    128
C** VARIABLE IS REFERENCED                                              FLOWCK    129
C** IF UNDEFINED OR ASSIGNED, ISSUE DIAGNOSTIC                         FLOWCK    130
   60 IF(IDTBL(2,LOC) .EQ. 0) GO TO 140                                 FLOWCK    131
      IF(IDTBL(2,LOC) .EQ. 3) GOTO 130                                  FLOWCK    132
C** SET "REFERENCED" FLAG                                               FLOWCK    133
      GO TO 500                                                         FLOWCK    134
C** VARIABLE IS A DO INDEX                                              FLOWCK    135
   70 IF(BITGET(IDTBL(3,LOC),13,1) .EQ. 1) GO TO 120                    FLOWCK    136
      IF(IDTBL(2,LOC) .EQ. 2) GO TO 75                                  FLOWCK    137
C** SET "DO INDEX" FLAG                                                 FLOWCK    138
   72 IDTBL(2,LOC)=2                                                    FLOWCK    139
      GO TO 500                                                         FLOWCK    140
C** FLAG INDICATES THAT VARIABLE WAS ONCE A DO INDEX                    FLOWCK    141
C** MAKE SURE THAT IT IS NOT CURRENTLY A DO INDEX                       FLOWCK    142
   75 IF(ILOOP .EQ. 0) GO TO 72                                         FLOWCK    143
   77 IF(LOC .EQ. ISTACK(4,ILOOP)) GO TO 110                            FLOWCK    144
      IF(ISTACK(3,ILOOP) .EQ. 0) GO TO 72                               FLOWCK    145
      ILOOP=ISTACK(3,ILOOP)                                             FLOWCK    146
      GO TO 77                                                          FLOWCK    147
C** VARIABLE IS ASSIGNED                                                FLOWCK    148
   80 IF(BITGET(IDTBL(3,LOC),13,1) .EQ. 1) GO TO 120                    FLOWCK    149
      IF(IDTBL(2,LOC) .EQ. 2) GO TO 85                                  FLOWCK    150
C** SET "ASSIGNED" FLAG                                                 FLOWCK    151
   82 IDTBL(2,LOC)=3                                                    FLOWCK    152
      GO TO 500                                                         FLOWCK    153
C** FLAG INDICATES THAT VARIABLE WAS ONCE A DO INDEX                    FLOWCK    154
C** MAKE SURE THAT IT IS NOT CURRENTLY A DO INDEX                       FLOWCK    155
   85 IF(ILOOP .EQ. 0) GO TO 82                                         FLOWCK    156
   87 IF(LOC .EQ. ISTACK(4,ILOOP)) GO TO 110                            FLOWCK    157
      IF(ISTACK(3,ILOOP) .EQ. 0) GO TO 82                               FLOWCK    158
      ILOOP=ISTACK(3,ILOOP)                                             FLOWCK    159
      GO TO 87                                                          FLOWCK    160
C** VARIABLE IS REFERENCED BY ASSIGNED GO TO                            FLOWCK    161
C** MAKE SURE VARIABLE WAS ASSIGNED                                     FLOWCK    162
   90 IF(IDTBL(2,LOC) .NE. 3) GO TO 150                                 FLOWCK    163
      GO TO 500                                                         FLOWCK    164
C** VARIABLE IS MADE UNDEFINED                                          FLOWCK    165
  100 IDTBL(2,LOC)=0                                                    FLOWCK    166
      GO TO 500                                                         FLOWCK    167
C** THIS LOOP CHECKS TO SEE IF A DO PARAMETER HAS BEEN REDEFINED        FLOWCK    168
  180 JLOOP=BITGET(IDTBL(3,LOC),36,18)                                  FLOWCK    169
  185 IF(ILOOP .EQ. JLOOP) GO TO 160                                    FLOWCK    170
```

```
      ILOOP=ISTACK(3,ILOOP)                                              FLOWCK    171
      IF(ILOOP .EQ. 0) GO TO 30                                          FLOWCK    172
      GO TO 185                                                          FLOWCK    173
C** VARIABLE IS A DO PARAMETER                                           FLOWCK    174
C** IF UNDEFINED OR ASSIGNED ISSUE DIAGNOSTIC                           FLOWCK    175
  200 IF(IDTBL(2,LOC) .EQ. 0) GO TO 140                                  FLOWCK    176
      IF(IDTBL(2,LOC) .EQ. 3) GO TO 130                                  FLOWCK    177
      IDTBL(2,LOC)=4                                                     FLOWCK    178
      GO TO 500                                                          FLOWCK    179
C** SET ERROR CODES                                                      FLOWCK    180
  110 IERC=67                                                            FLOWCK    181
      GO TO 400                                                          FLOWCK    182
  120 IERC=68                                                            FLOWCK    183
      GO TO 400                                                          FLOWCK    184
  130 IERC=69                                                            FLOWCK    185
      GO TO 400                                                          FLOWCK    186
  140 IERC=70                                                            FLOWCK    187
      GO TO 400                                                          FLOWCK    188
  150 IERC=71                                                            FLOWCK    189
      GO TO 400                                                          FLOWCK    190
  160 IERC=72                                                            FLOWCK    191
  400 IRX=1                                                              FLOWCK    192
C** SET ERROR FLAG AND ISSUE DIAGNOSTIC                                  FLOWCK    193
      IF(BITGET(IDTBL(3,LOC),15,1) .EQ. 1) GO TO 500                     FLOWCK    194
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),1,15)                             FLOWCK    195
      CALL ERROR(IERC,IDTBL(1,LOC))                                      FLOWCK    196
      IF(IERC .NE. 70 .OR. NPATH .EQ. 0) GO TO 500                       FLOWCK    197
      WRITE(6,410) (IPATH(K),K=1,NPATH)                                  FLOWCK    198
  410 FORMAT(6X,15H ALONG THE PATH,(10I6))                               FLOWCK    199
  500 CONTINUE                                                           FLOWCK    200
 1000 CONTINUE                                                           FLOWCK    201
C** SCANNING OF THIS PATH COMPLETE - GET NEXT PATH                       FLOWCK    202
C** STARTING FROM BOTTOM OF FLOW PATH, FIND FIRST NEGATIVE NUMBER        FLOWCK    203
 1500 IF(FLWLST(NFLOW) .GT. 0) GO TO 1600                                FLOWCK    204
C** NEGATIVE NUMBER FOUND - TAKE NEXT BRANCH FROM TOP OF STACK           FLOWCK    205
      IBLKST=IABS(ISTCK(NSTCK))                                          FLOWCK    206
      IF(ISTCK(NSTCK) .LT. 0) FLWLST(NFLOW)=-FLWLST(NFLOW)               FLOWCK    207
      NSTCK=NSTCK-1                                                      FLOWCK    208
      NOC=0                                                              FLOWCK    209
      GO TO 5                                                            FLOWCK    210
 1600 NFLOW=NFLOW-1                                                      FLOWCK    211
      IF(NFLOW .GT. 0) GO TO 1500                                        FLOWCK    212
C** NO MORE BRANCHES LEFT                                                FLOWCK    213
      IF(NLABEL .EQ. 0) GO TO 2010                                       FLOWCK    214
C** THIS LOOP CHECKS TO SEE IF STATEMENT LABELS WERE ALL REFERENCED      FLOWCK    215
      DO 2000 J=1,NLABEL                                                 FLOWCK    216
      IF(BITGET(STATRA(2,J),6,6) .EQ. 28) GO TO 2000                     FLOWCK    217
      IF(BITGET(STATRA(2,J),18,3) .EQ. 1) GO TO 2000                     FLOWCK    218
      WRITE(6,1800) STATRA(1,J)                                          FLOWCK    219
      IRX=1                                                              FLOWCK    220
 1800 FORMAT(6X,57H THERE IS NO COMPLETE PATH THAT CONTAINS STATEMENT NU  FLOWCK    221
     1MBER,I6)                                                           FLOWCK    222
 2000 CONTINUE                                                           FLOWCK    223
 2010 IF(IRX .EQ. 0) WRITE(6,2020)                                       FLOWCK    224
 2020 FORMAT(//6X,16H NO ERRORS FOUND)                                   FLOWCK    225
      WRITE(6,2100) NPTHS                                                FLOWCK    226
 2100 FORMAT(//////6X,25H NUMBER OF PATHS CHECKED-,I6)                   FLOWCK    227
      CLTM2=SECOND(T)                                                    FLOWCK    228
      TOTTIM=CLTM2-CLTM1                                                 FLOWCK    229
      WRITE(6,2700) TOTTIM                                               FLOWCK    230
 2700 FORMAT(//46X,20H FLOW ANALYSIS TOOK ,F8.3,11H CP SECONDS)          FLOWCK    231
      RETURN                                                             FLOWCK    232
 3000 WRITE(6,3001)                                                      FLOWCK    233
 3001 FORMAT(//31X,57H FLOW ANALYSIS WAS NOT PERFORMED DUE TO ERRORS IN  FLOWCK    234
     $PROGRAM)                                                           FLOWCK    235
      IFL=IRP+1                                                          FLOWCK    236
      RETURN                                                             FLOWCK    237
 4000 WRITE(6,4001)                                                      FLOWCK    238
 4001 FORMAT(//29X,63H TABLE OVERFLOW DURING FLOW ANALYSIS - FLOW ANALYS FLOWCK    239
     $IS TERMINATED)                                                     FLOWCK    240
      RETURN                                                             FLOWCK    241
 5000 WRITE(6,5001)                                                      FLOWCK    242
 5001 FORMAT(//29X,63H STACK OVERFLOW DURING FLOW ANALYSIS - FLOW ANALYS FLOWCK    243
     $IS TERMINATED)                                                     FLOWCK    244
      RETURN                                                             FLOWCK    245
      END                                                                FLOWCK    246
```

```
      SUBROUTINE FNCSTR                                                 FNCSTR     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH       2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A      80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH       4
      COMMON/FUNC/IFNCRA(5,12),MARGS,IARGS(50),FNCLOC(5),NFUNC          CY58A      37
      COMMON/LIST/NLIST,NINTFC,ISUBLT(2,200),INTFAC(300)                FNCSTR     5
      INTEGER FNCLOC,BITPUT,BITGET                                      FNCSTR     6
C** THIS ROUTINE IS CALLED AFTER PARSING AN EXPRESSION, TO PROCESS       FNCSTR     7
C** ALL FUNCTION REFERENCES IN THE EXPRESSION                           FNCSTR     8
      IF(NFUNC .EQ. 0) RETURN                                           FNCSTR     9
      DO 40 I=1,NFUNC                                                   FNCSTR    10
C** GET SYMBOL TABLE LOCATION OF NEXT FUNCTION                          FNCSTR    11
      LOC=FNCLOC(I)                                                     FNCSTR    12
C** SKIP IF STATEMENT FUNCTION                                          FNCSTR    13
      IF(BITGET(IDTBL(3,LOC),19,1) .EQ. 1) GO TO 50                     FNCSTR    14
C** GET NUMBER OF ARGUMENTS                                             FNCSTR    15
      NARG=IFNCRA(I,1)                                                  FNCSTR    16
      IVAR=0                                                            FNCSTR    17
C** GET FUNCTION TYPE                                                   FNCSTR    18
      ITP=BITGET(IDTBL(3,LOC),10,3)                                     FNCSTR    19
      IF(BITGET(IDTBL(3,LOC),18,1) .EQ. 1) GO TO 20                     FNCSTR    20
C** FUNCTION NAME HAS NOT YET APPEARED IN PROGRAM - SET FLAG            FNCSTR    21
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),1,18)                            FNCSTR    22
C** SEARCH SESCOMP LIST FOR NAME                                        FNCSTR    23
      DO 5 J=1,NLIST                                                    FNCSTR    24
      IF(IDTBL(1,LOC) .NE. ISUBLT(1,J)) GO TO 5                         FNCSTR    25
C** NAME FOUND - STORE SESCOMP LIST LOCATION IN SYMBOL TABLE            FNCSTR    26
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),J,36)                            FNCSTR    27
      LISTLC=J                                                          FNCSTR    28
      GO TO 21                                                          FNCSTR    29
    5 CONTINUE                                                          FNCSTR    30
C** NAME NOT FOUND IN SESCOMP LIST - ISSUE DIAGNOSTIC                   FNCSTR    31
      CALL ERROR(52)                                                    FNCSTR    32
C** STORE NAME IN SESCOMP LIST                                          FNCSTR    33
      NLIST=NLIST+1                                                     FNCSTR    34
      IF(NLIST .GT. 200) GO TO 60                                       CY58A     38
      ISUBLT(1,NLIST)=IDTBL(1,LOC)                                      FNCSTR    35
C** STORE LIST LOCATION IN SYMBOL TABLE                                 FNCSTR    36
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),NLIST,36)                        FNCSTR    37
C** INCREMENT INTERFACE DEFINITION TABLE POINTER                        FNCSTR    38
      IPTR=NINTFC+1                                                     FNCSTR    39
      IF(ITYP .EQ. 8 .AND. I .EQ. 1) ITP=0                              FNCSTR    40
C** STORE INTERFACE DEFINITION TABLE POINTER AND FUNCTION TYPE IN        FNCSTR    41
C** SESCOMP LIST                                                        FNCSTR    42
      ISUBLT(2,NLIST)=BITPUT(IPTR,ITP,13)                               FNCSTR    43
C** STORE NO. OF ARGUMENTS IN SESCOMP LIST                              FNCSTR    44
      ISUBLT(2,NLIST)=BITPUT(ISUBLT(2,NLIST),NARG,6)                    FNCSTR    45
C** UPDATE INTERFACE DEFINITION TABLE COUNTER                           FNCSTR    46
      NINTFC=IPTR+(NARG-1)/6                                            FNCSTR    47
      IF(NINTFC .GT. 300) GO TO 70                                      CY58A     39
C** STORE INTERFACE DEFINITION                                          FNCSTR    48
      DO 10 J=IPTR,NINTFC                                               FNCSTR    49
   10 INTFAC(J)=IFNCRA(I,J-IPTR+2)                                      FNCSTR    50
      GO TO 40                                                          FNCSTR    51
C** FUNCTION NAME HAS PREVIOUSLY OCCURED - GET SESCOMP LIST LOCATION    FNCSTR    52
   20 LISTLC=BITGET(IDTBL(3,LOC),36,9)                                  FNCSTR    53
C** GET INTERFACE DEFINITION TABLE POINTER                              FNCSTR    54
```

```
   21 IPTR=BITGET(ISUBLT(2,LISTLC),60,15)                              FNCSTR     55
      IF(BITGET(ISUBLT(2,LISTLC),14,1) .EQ. 1) GO TO 22                FNCSTR     56
C** GET NO. OF ARGUMENTS AND CHECK VALIDITY                            FNCSTR     57
      NAR2=BITGET(ISUBLT(2,LISTLC),6,6)                                FNCSTR     58
      IF(NARG .NE. NAR2) CALL ERROR(26,IDTBL(1,LOC))                   FNCSTR     59
      NARGS=MIN0(NARG,NAR2)                                            FNCSTR     60
      GO TO 24                                                         FNCSTR     61
C** VARIABLE NO. OF ARGUMENTS - SET FLAG                               FNCSTR     62
   22 IVAR=1                                                           FNCSTR     63
      IF(NARG .LT. 2) CALL ERROR(26,IDTBL(1,LOC))                      FNCSTR     64
      NARGS=NARG                                                       FNCSTR     65
C** SET ARGUMENT TYPE AND DIMENSIONALITY                               FNCSTR     66
      ITP1=BITGET(INTFAC(IPTR),3,3)                                    FNCSTR     67
      NDIM1=BITGET(INTFAC(IPTR),6,3)                                   FNCSTR     68
   24 IF(ITYP .EQ. 8 .AND. I .EQ. 1) GO TO 25                          FNCSTR     69
      IF(BITGET(ISUBLT(2,LISTLC),10,4) .EQ. 4) GO TO 25                FNCSTR     70
C** CHECK TYPES OF INTRINSIC FUNCTIONS                                 FNCSTR     71
      JTP=BITGET(ISUBLT(2,LISTLC),13,3)                                FNCSTR     72
      IF(JTP .NE. ITP) CALL ERROR(49,IDTBL(1,LOC))                     FNCSTR     73
C** SET INTERFACE DEFINITION TABLE POINTER                             FNCSTR     74
   25 NDPTR=IPTR+(NARGS-1)/6                                           FNCSTR     75
      KOUNT=0                                                          FNCSTR     76
C** THESE TWO LOOPS CHECK THE ARGUMENT LIST AGAINST THE                FNCSTR     77
C** INTERFACE DEFINITION                                               FNCSTR     78
      DO 32 K=IPTR,NDPTR                                               FNCSTR     79
      ICOL1=-6                                                         FNCSTR     80
      ICOL2=-3                                                         FNCSTR     81
      DO 32 J=1,6                                                      FNCSTR     82
      KOUNT=KOUNT+1                                                    FNCSTR     83
      IF(KOUNT .GT. NARGS) GO TO 40                                    FNCSTR     84
      ICOL1=ICOL1+9                                                    FNCSTR     85
      ICOL2=ICOL2+9                                                    FNCSTR     86
      IF(IVAR .EQ. 1) GO TO 26                                         FNCSTR     87
C** GET ARGUMENT TYPE AND DIMENSIONALITY FROM SYMBOL TABLE              FNCSTR     88
      ITP1=BITGET(INTFAC(K),ICOL1,3)                                   FNCSTR     89
      NDIM1=BITGET(INTFAC(K),ICOL2,3)                                  FNCSTR     90
   26 ITP2=BITGET(IFNCRA(I,K-IPTR+2),ICOL1,3)                          FNCSTR     91
      NDIM2=BITGET(IFNCRA(I,K-IPTR+2),ICOL2,3)                         FNCSTR     92
C** CHECK DIMENSIONALITY AND TYPE FOR VALIDITY                         FNCSTR     93
      IF(NDIM1 .NE. NDIM2) CALL ERROR(50,KOUNT)                        FNCSTR     94
      IF(ITP2 .EQ. 0) GO TO 32                                         FNCSTR     95
      IF(ITP1 .EQ. 0) GO TO 28                                         FNCSTR     96
      IF(ITP1 .NE. ITP2) CALL ERROR(51,KOUNT)                          FNCSTR     97
      GO TO 32                                                         FNCSTR     98
   28 INTFAC(K)=BITPUT(INTFAC(K),ITP2,ICOL1)                           FNCSTR     99
   32 CONTINUE                                                         FNCSTR    100
      GO TO 40                                                         FNCSTR    101
C** STATEMENT FUNCTION - CALL STATEMENT FUNCTION PROCESSOR             FNCSTR    102
   50 CALL STFNC(I)                                                    CY58A      40
   40 CONTINUE                                                         FNCSTR    104
      RETURN                                                           FNCSTR    105
   60 CALL ERROR(92)                                                   CY58A      41
      STOP                                                             CY58A      42
   70 CALL ERROR(93)                                                   CY58A      43
      STOP                                                             CY58A      44
      END                                                              FNCSTR    106
```

FORTRAN Version

```
      SUBROUTINE FORM                                                    FORM2  2
      COMMON/LVARGS/LVFUNC,LVVARG,LVVAD,LVVPOS,LVVTYP,       LVVAL,
     +LVHEAD,LVVNVL,LVDEST,LVVALS(10),LVTYPE(10),LVSKIP
      COMMON/LVTABL/LVTSIZ,LVMAP(    1)/LVVSEQ/LVSIZE,LVSQSP(     1)
      COMMON /HL/ HOL,ACTION,FUNC1,FUNC2,FUNC3,LEFT,RIGHT,STRING         FORM2  3
      COMMON /VAR/ VFOR,NCHAR,NCHARP,CHAR,NDICT                          FORM2  4
      COMMON /TYP/ NARRAY,TYPE1,TYPE2,ERRFLG                             FORM2  5
      COMMON /STRING/ NTYPE,NSTR,STR                                     FORM2  6
      INTEGER BITPUT,BITGET                                              FORM2  7
      INTEGER VFOR(15),CHAR,STR(1)                                       FORM2  8
      LOGICAL ERRFLG                                                     FORM2  9
      GO TO 25000
25001 CONTINUE
      IF(CHAR .NE. 1HX) NDICT=-NDICT                                     FORM2 11
      NCHARP=NCHARP+1                                                    FORM2 12
      STR(NCHARP)=NDICT                                                  FORM2 13
      IF(.NOT. ERRFLG) RETURN                                            FORM2 14
      NCHAR=NCHAR+1                                                      FORM2 15
      NC=1+(NCHARP-1)/8                                                  FORM2 16
      ICHAR=BITGET(CHAR,6,6)                                             FORM2 17
      VFOR(NC)=BITPUT(VFOR(NC),ICHAR,6*NCHAR)                            FORM2 18
      IF(NCHAR .EQ. 8) NCHAR=0                                           FORM2 19
      RETURN
25000 CONTINUE
      GO TO 25001
      END
```

GIRL Version

```
$     SUBROUTINE FORM                                                    FORM2    2
      COMMON /HL/ HOL,ACTION,FUNC1,FUNC2,FUNC3,LEFT,RIGHT,STRING         FORM2    3
      COMMON /VAR/ VFOR,NCHAR,NCHARP,CHAR,NDICT                          FORM2    4
      COMMON /TYP/ NARRAY,TYPE1,TYPE2,ERRFLG                             FORM2    5
      COMMON /STRING/ NTYPE,NSTR,STR                                     FORM2    6
      INTEGER BITPUT,BITGET                                              FORM2    7
      INTEGER VFOR(15),CHAR,STR(1)                                       FORM2    8
      LOGICAL ERRFLG                                                     FORM2    9
G     EXECUTE                                                            FORM2   10
      IF(CHAR .NE. 1HX) NDICT=-NDICT                                     FORM2   11
      NCHARP=NCHARP+1                                                    FORM2   12
      STR(NCHARP)=NDICT                                                  FORM2   13
      IF(.NOT. ERRFLG) RETURN                                            FORM2   14
      NCHAR=NCHAR+1                                                      FORM2   15
      NC=1+(NCHARP-1)/8                                                  FORM2   16
      ICHAR=BITGET(CHAR,6,6)                                             FORM2   17
      VFOR(NC)=BITPUT(VFOR(NC),ICHAR,6*NCHAR)                            FORM2   18
      IF(NCHAR .EQ. 8) NCHAR=0                                           FORM2   19
G     COMPLETE                                                           FORM2   20
```

```
      SUBROUTINE FORMEL                                                 FORM     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH     2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A   80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH     4
      INTEGER B(50),D,BITGET                                            FORM     4
C** THIS ROUTINE IS CALLED BY "GNLE" TO PROCESS LANGUAGE ELEMENTS       FORM     5
      GO TO(100,10,20,30,40,50,100,100),JTYP                            FORM     6
C** PACK NAME INTO SINGLE WORD AND STORE IN "NXTID"                     FORM     7
   10 CALL CAA(D,M,NXTID)                                               FORM     8
      RETURN                                                            FORM     9
   20 DO 25 I=1,10                                                      FORM    10
      IF(D(I) .NE. 1HH) GO TO 25                                        FORM    11
C** GET SIZE OF HOLLERITH STRING                                        FORM    12
      CALL CAI(D,I-1,N2)                                                FORM    13
      IF(N2 .LT. 1) GO TO 110                                           FORM    14
      M=N2+I                                                            FORM    15
      IF(M .GT. 500) CALL ERROR(5)                                      FORM    16
      JPTR=LSTART+M                                                     FORM    17
C** CHECK FOR NON-STANDARD CHARACTER IN STRING                          FORM    18
      IST=I+1                                                           FORM    19
      DO 22 J=IST,M                                                     FORM    20
      ICHAR=BITGET(D(J),6,6)                                            FORM    21
      IF(ICHAR .GT. 57B) GO TO 120                                      FORM    22
   22 CONTINUE                                                          FORM    23
      IF(ITYP .EQ. 28) RETURN                                           FORM    24
      IF(N2 .GT. 4) CALL ERROR(5)                                       FORM    25
      RETURN                                                            FORM    26
   25 CONTINUE                                                          FORM    27
      CALL ERROR(3)                                                     FORM    28
      RETURN                                                            FORM    29
C** CHECK VALIDITY OF REAL NUMBER                                       FORM    30
   30 CALL CAR(D,M,IDES)                                                FORM    31
      RETURN                                                            FORM    32
C** CHECK VALIDITY OF INTEGER AND STORE VALUE IN "N2"                   FORM    33
   40 CALL CAI(D,M,N2)                                                  FORM    34
      RETURN                                                            FORM    35
C** CHECK VALIDITY OF COMPLEX NUMBER                                    FORM    36
   50 DO 55 I=2,M                                                       FORM    37
      IF(D(I) .EQ. 1H,) GO TO 60                                        FORM    38
   55 B(I-1)=D(I)                                                       FORM    39
   60 CALL CAR(B,I-2,IDES)                                              FORM    40
      M2=M-I-1                                                          FORM    41
      DO 65 J=1,M2                                                      FORM    42
   65 B(J)=D(I+J)                                                       FORM    43
      CALL CAR(B,M2,IDES)                                               FORM    44
  100 RETURN                                                            FORM    45
  110 CALL ERROR(7)                                                     FORM    46
      RETURN                                                            FORM    47
  120 CALL ERROR(23)                                                    FORM    48
      RETURN                                                            FORM    49
      END                                                               FORM    50
```

```
      SUBROUTINE FRMAT                                                  FRMAT      2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH       2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A     80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH       4
      COMMON/FORMAT/IDESST,IDESND,IGPST,IGPND,IGRP,SEPST,SEPND,         CY58A      2
     1 DIR,ICOM,ISEP                                                    FRMAT      5
      DIMENSION RPLOC(20),IALPH(6)                                      FRMAT      6
      INTEGER A,RPLOC,AICH,RPAR,BLANK,SEPST,SEPND,DIR                   FRMAT      7
      DATA BLANK/1H /,AICH/1HH/,LPAR/1H(/,RPAR/1H)/                     FRMAT      8
      DATA (IALPH(I),I=1,6)/1HF,1HO,1HR,1HM,1HA,1HT/                    FRMAT      9
C** THIS ROUTINE PROCESSES FORMAT STATEMENTS AND RETURNS                FRMAT     10
C**      IFRMT=1 - VALID     IFRMAT=0 - INVALID                         FRMAT     11
      IFRMT=0                                                           FRMAT     12
C** CHECK SPELLING                                                      FRMAT     13
      DO 4 I=1,6                                                        FRMAT     14
      IF(NEXT(JPTR) .NE. IALPH(I)) GO TO 70                             FRMAT     15
    4 CONTINUE                                                          FRMAT     16
      NSTART=JPTR                                                       FRMAT     17
      IF(NEXT(JPTR) .NE. LPAR) GO TO 70                                 FRMAT     18
      DO 10 I=1,N                                                       FRMAT     19
      IF(ITYPE(JPTR) .EQ. 2) GO TO 1                                    FRMAT     20
      IF(JPTR .GT. N) GO TO 12                                          FRMAT     21
      GO TO 10                                                          FRMAT     22
    1 JPTR=JPTR-1                                                       FRMAT     23
      CALL GNLE                                                         FRMAT     24
      IF(JTYP .NE. 3) GO TO 10                                          FRMAT     25
      J1=JPTR-1                                                         FRMAT     26
      IH=0                                                              FRMAT     27
C** PUT BLANKS IN HOLLERITH STRINGS                                     FRMAT     28
      DO 5 J=LSTART,J1                                                  FRMAT     29
      IF(IH .EQ. 1) GO TO 3                                             FRMAT     30
      IF(A(J) .EQ. AICH) IH=1                                           FRMAT     31
      GO TO 5                                                           FRMAT     32
    3 A(J)=BLANK                                                        FRMAT     33
    5 CONTINUE                                                          FRMAT     34
   10 CONTINUE                                                          FRMAT     35
   12 NPAR=0                                                            FRMAT     36
      NRP=0                                                             FRMAT     37
      DO 20 I=NSTART,N                                                  FRMAT     38
      IF(A(I) .NE. LPAR) GO TO 15                                       FRMAT     39
      NPAR=NPAR+1                                                       FRMAT     40
      IF(NPAR .GT. 3) GO TO 70                                          FRMAT     41
      GO TO 20                                                          FRMAT     42
   15 IF(A(I) .NE. RPAR) GO TO 20                                       FRMAT     43
      NPAR=NPAR-1                                                       FRMAT     44
      NRP=NRP+1                                                         FRMAT     45
C** STORE LOCATIONS OF RIGHT PARENS                                     FRMAT     46
      RPLOC(NRP)=I                                                      FRMAT     47
      IF(NPAR .LT. 0) GO TO 70                                          FRMAT     48
   20 CONTINUE                                                          FRMAT     49
      IF(NPAR .NE. 0) GO TO 70                                          FRMAT     50
      IF(NEXT(RPLOC(NRP)+1) .NE. BLANK) GO TO 70                        FRMAT     51
      DO 60 I=1,NRP                                                     FRMAT     52
      IGPND=RPLOC(I)                                                    FRMAT     53
      DO 25 J=1,N                                                       FRMAT     54
      K=IGPND-J                                                         FRMAT     55
      IF(A(K) .NE. LPAR) GO TO 25                                       FRMAT     56
```

```
C** GET CORRESPONDING LEFT PAREN FOR RIGHT PAREN                    FRMAT     57
      IGPST=K                                                       FRMAT     58
      GO TO 30                                                      FRMAT     59
   25 CONTINUE                                                      FRMAT     60
C** CHECK SYNTAX OF GROUP                                           FRMAT     61
   30 CALL GROUP                                                    FRMAT     62
      IF(IGRP .EQ. 0) RETURN                                        FRMAT     63
      IF(I .EQ. NRP) GO TO 65                                       FRMAT     64
      JPTR=IGPST-1                                                  FRMAT     65
   31 CONTINUE                                                      FRMAT     66
      IF(IPREV(JPTR) .EQ. 2) GO TO 31                               FRMAT     67
      IGPST=JPTR+2                                                  FRMAT     68
      SEPST=IGPND+1                                                 FRMAT     69
C** CHECK NEXT SEPARATOR                                            FRMAT     70
      DIR=1                                                         FRMAT     71
      CALL SEPAR                                                    FRMAT     72
      IF(ISEP .NE. 1) GO TO 40                                      FRMAT     73
      IGPND=SEPND                                                   FRMAT     74
      GO TO 50                                                      FRMAT     75
   40 IF(NEXT(SEPST) .NE. RPAR) GO TO 70                            FRMAT     76
C** CHECK PRECEDING SEPARATOR                                       FRMAT     77
      SEPST=IGPST-1                                                 FRMAT     78
      DIR=-1                                                        FRMAT     79
      CALL SEPAR                                                    FRMAT     80
      IF(ISEP .NE. 1) GO TO 45                                      FRMAT     81
      IGPST=SEPND                                                   FRMAT     82
      GO TO 50                                                      FRMAT     83
   45 IF(A(SEPND) .NE. LPAR) GO TO 70                               FRMAT     84
   50 DO 55 J=IGPST,IGPND                                           FRMAT     85
      A(J)=BLANK                                                    FRMAT     86
   55 CONTINUE                                                      FRMAT     87
   60 CONTINUE                                                      FRMAT     88
   65 IFRMT=1                                                       FRMAT     89
      RETURN                                                        FRMAT     90
   70 CALL ERROR(7)                                                 FRMAT     91
      RETURN                                                        FRMAT     92
      END                                                           FRMAT     93
```

```
      SUBROUTINE GENROL                                                 GENROL     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH       2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A     80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IOES                                  RICH       4
      COMMON/GLOBAL/NBLK,NREF,NSUBS,BLKTBL(200),EXTTBL(100),ISUBS(100)  GENROL     4
      COMMON/LIST/NLIST,NINTFC,ISUBLT(2,200),INTFAC(300)                GENROL     5
      COMMON/INPOUT/NCALL,IN,IOP                                        GENROL     6
      COMMON/WASTE/IDUM(63)                                             GENROL     7
      INTEGER BLKTBL,EXTTBL,BITGET                                      GENROL     8
C** THIS ROUTINE IS CALLED IN THE ROLL CALL MODE TO GENERATE THE MAIN    GENROL     9
C** ROLL CALL PROGRAM                                                   GENROL    10
      WRITE(6,2)                                                        GENROL    11
    2 FORMAT(1H1)                                                       GENROL    12
C** GENERATE PROGRAM CARD                                               GENROL    13
      WRITE(IOP,4)                                                      GENROL    14
    4 FORMAT(5X,48H PROGRAM ROLCAL(OUTPUT,TAPE6=OUTPUT,TAPE3,TAPE9,/    GENROL    15
     * 5X,44H* TAPE10,TAPE11,TAPE12,TAPE13,TAPE14,TAPE15))              GENROL    16
      IF(NBLK .EQ. 0) GO TO 6                                           GENROL    17
      K=-1                                                              GENROL    18
C** THIS LOOP GENERATES COMMON STATEMENTS                               GENROL    19
      DO 5 I=1,NBLK                                                     GENROL    20
      K=K+1                                                             GENROL    21
      INDEX=BLKTBL(I)                                                   GENROL    22
      ISZ=BITGET(ISUBLT(2,INDEX),30,15)                                 GENROL    23
      WRITE(IOP,3) ISUBLT(1,INDEX),K,ISZ                                GENROL    24
    3 FORMAT(5X,8H COMMON/,A8,3H/IX,I2,1H(,I6,1H))                      GENROL    25
    5 CONTINUE                                                          GENROL    26
C** GENERATE LOOP TO DUMMY OUT COMMON BLOCKS                            GENROL    27
    6 WRITE(IOP,7) MODE                                                 GENROL    28
    7 FORMAT(5X,4H J=1/5X,6H MODE=,I1/5X,10H REWIND 13/5X,10H REWIND 14/ GENROL    29
     $ 5X,10H REWIND 15/5X,13H DO 10 I=1,13/5X,6H J=J-1)                GENROL    30
      IF(NBLK .EQ. 0) GO TO 22                                          GENROL    31
      K=-1                                                              GENROL    32
      DO 20 I=1,NBLK                                                    GENROL    33
      K=K+1                                                             GENROL    34
      KK=1000+K                                                         GENROL    35
      INDEX=BLKTBL(I)                                                   GENROL    36
      ISZ=BITGET(ISUBLT(2,INDEX),30,15)                                 GENROL    37
      WRITE(IOP,10) KK,ISZ,K,KK                                         GENROL    38
   10 FORMAT(5X,4H DO ,I4,5H K=1,,I6/5X,3H IX,I2,5H(K)=1/1X,I4,         GENROL    39
     $ 9H CONTINUE)                                                     GENROL    40
      IF(ISUBLT(1,INDEX) .NE. 6HSESCOM) GO TO 20                        GENROL    41
C** SET I/O DEVICES IN COMMON BLOCK SESCOM                              GENROL    42
      WRITE(IOP,15) K,K,K                                               GENROL    43
   15 FORMAT(5X,3H IX,I2,7H(17)=10/5X,3H IX,I2,7H(20)=11/               GENROL    44
     * 5X,3H IX,I2,7H(23)=12)                                          GENROL    45
   20 CONTINUE                                                          GENROL    46
C** GENERATE CALL TO THE MODULE AND DUMMY ARGUMENT LIST - MODULE        GENROL    47
C** CONTAINS CALLS TO "ROLCHK"                                          GENROL    48
   22 NARG=BITGET(IDTBL(3,1),7,6)                                       GENROL    49
      DO 30 I=1,NARG                                                    GENROL    50
      IF(I .EQ. NARG) GO TO 25                                          GENROL    51
      IDUM(I)=2H0,                                                      GENROL    52
      GO TO 30                                                          GENROL    53
   25 IDUM(I)=2HJ)                                                      GENROL    54
   30 CONTINUE                                                          GENROL    55
      WRITE(IOP,35) (IDTBL(1,1),(IDUM(I),I=1,NARG))                     GENROL    56
   35 FORMAT(5X,6H CALL ,A6,1H(,41A2/5X,1H1,1X,22A2)                    GENROL    57
C** GENERATE REMAINDER OF PROGRAM INCLUDING CALLS TO ROLL CALL          GENROL    58
C** AUXILIARY PROGRAMS "MODID" AND "COMPARE"                            GENROL    59
      WRITE(IOP,40)                                                     GENROL    60
   40 FORMAT(5X,23H IF(MODE .EQ. 3)GO TO 5/5X,14H CALL MODID(J)/        GENROL    61
     $ 3X,12H 5 ENDFILE 3/2X,12H 10 CONTINUE/5X,12H CALL CMPARE/        GENROL    62
     $ 5X,9H REWIND 3/5X,10H REWIND 13/5X,10H REWIND 14/5X,10H REWIND 15 GENROL    63
     $ /5X,5H STOP/5X,4H END)                                           GENROL    64
      RETURN                                                            GENROL    65
      END                                                               GENROL    66
```

```
      SUBROUTINE GLOTAB                                                 GLOTAB     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH      2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A     80
     2 LTYP,ITYP,IBLKOT,MODE,IERR,IDES                                  RICH      4
      COMMON/GLOBAL/NBLK,NREF,NSUBS,BLKTBL(200),EXTTBL(100),ISUBS(100)  GLOTAB     4
      COMMON/LIST/NLIST,NINTFC,ISUBLT(2,200),INTFAC(300)                GLOTAB     5
      INTEGER BITGET,BLKTBL,EXTTBL,KLAS(2,7)                            GLOTAB     6
      DATA KLAS/10HUSER SUPPL,3HIED,10HSUBROUTINE,7H MODULE,            GLOTAB     7
     * 10HFUNCTION M,5HODULE,9HANCILLARY,10HSUBPROGRAM,10HANSI FUNCT,   GLOTAB     8
     * 3HION,10HMAIN PROGR,2HAM,10HEXTRAORDIN,10HARY SUBPR./            GLOTAB     9
C** THIS ROUTINE DISPLAYS THE GLOBAL REFERENCE TABLE                    GLOTAB    10
      WRITE(6,1)                                                        GLOTAB    11
    1 FORMAT(1H1,48X,23H GLOBAL REFERENCE TABLE)                        GLOTAB    12
      IF(NREF .EQ. 0) GO TO 25                                          GLOTAB    13
C** DISPLAY EXTERNAL REFERENCES                                         GLOTAB    14
      WRITE(6,2)                                                        GLOTAB    15
    2 FORMAT(//50X,20H EXTERNAL REFERENCES)                             GLOTAB    16
      DO 20 I=1,NREF                                                    GLOTAB    17
C** GET SESCOMP LIST LOCATION                                           GLOTAB    18
      INDEX=EXTTBL(I)                                                   GLOTAB    19
C** SET SUBPROGRAM CLASS                                                GLOTAB    20
      J=BITGET(ISUBLT(2,INDEX),10,4)                                    GLOTAB    21
      WRITE(6,10) ISUBLT(1,INDEX),KLAS(1,J+1),KLAS(2,J+1)               GLOTAB    22
   10 FORMAT(45X,A6,4X,2A10)                                            GLOTAB    23
   20 CONTINUE                                                          GLOTAB    24
   25 IF(NBLK .EQ. 0 .OR. (NBLK .EQ. 1 .AND. ISUBLT(1,BLKTBL(1)) .EQ.   GLOTAB    25
     $ 1H )) GO TO 40                                                   GLOTAB    26
C** DISPLAY LABELLED COMMON BLOCKS                                      GLOTAB    27
      WRITE(6,30)                                                       GLOTAB    28
   30 FORMAT(//49X,23H LABELLED COMMON BLOCKS/43X,11H BLOCK NAME,7X,    GLOTAB    29
     $ 5H SIZE,7X,6H CLASS)                                             GLOTAB    30
      DO 38 J=1,NBLK                                                    GLOTAB    31
C** GET SESCOMP LIST LOCATION                                           GLOTAB    32
      INDEX=BLKTBL(J)                                                   GLOTAB    33
C** GET COMMON BLOCK CATEGORY                                           GLOTAB    34
      ICAT=BITGET(ISUBLT(2,INDEX),10,4)-6                               GLOTAB    35
      ISZ=BITGET(ISUBLT(2,INDEX),30,15)                                 GLOTAB    36
   35 FORMAT(46X,A6,5X,I8,5X,9HCATEGORY ,I2)                            GLOTAB    37
      WRITE(6,35) ISUBLT(1,INDEX),ISZ,ICAT                              GLOTAB    38
   38 CONTINUE                                                          GLOTAB    39
C** DISPLAY SUBROUTINES                                                 GLOTAB    40
   40 WRITE(6,45)                                                       GLOTAB    41
   45 FORMAT(///48X,24H SUBROUTINES ENCOUNTERED)                        GLOTAB    42
      DO 60 I=1,NSUBS                                                   GLOTAB    43
C** GET SUBROUTINE CLASS                                                GLOTAB    44
      INDEX=ISUBS(I)                                                    GLOTAB    45
      J=BITGET(ISUBLT(2,INDEX),10,4)                                    GLOTAB    46
      WRITE(6,10) ISUBLT(1,INDEX),KLAS(1,J+1),KLAS(2,J+1)               GLOTAB    47
   60 CONTINUE                                                          GLOTAB    48
      RETURN                                                            GLOTAB    49
      END                                                               GLOTAB    50
```

```
      SUBROUTINE GNLE                                                   GNLE     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH     2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A   80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH     4
      COMMON/LOGIC/LOG,LOGST                                            GNLE     4
      COMMON/REALNO/IREAL,IRELNO,IP                                     GNLE     5
      INTEGER A,D,BLANK,PLUS,EQUALS,SLASH,RPAR,COMMA,ASTRIK,            GNLE     6
     1 AICH,DECPT                                                       GNLE     7
      DATA BLANK/1H /,PLUS/1H+/,MINUS/1H-/,EQUALS/1H=/,SLASH/1H//,      GNLE     8
     1 RPAR/1H)/,COMMA/1H,/,ASTRIK/1H*/,AICH/1HH/,LPAR/1H(/,DECPT/1H./  GNLE     9
C** THIS ROUTINE SCANS THE INPUT STRING STARTING AT "JPTR" AND RETURNS   GNLE    10
C** THE NEXT LANGUAGE ELEMENT                                           GNLE    11
C**       JTYP=0 - BLANK                                                GNLE    12
C**       JTYP=1 - ARITHMETIC OPERATOR                                  GNLE    13
C**       JTYP=2 - NAME                                                 GNLE    14
C**       JTYP=3 - HOLLERITH STRING                                     GNLE    15
C**       JTYP=4 - REAL NUMBER                                          GNLE    16
C**       JTYP=5 - INTEGER                                              GNLE    17
C**       JTYP=6 - COMPLEX NUMBER                                       GNLE    18
C**       JTYP=7 - LOGICAL OPERATOR OR CONSTANT                         GNLE    19
C**       JTYP=8 - INVALID                                              GNLE    20
      JTYP=0                                                            GNLE    21
      NXT=NEXT(JPTR)                                                    GNLE    22
      IF(NXT .EQ. BLANK) RETURN                                         GNLE    23
      LSTART=JPTR-1                                                     GNLE    24
      IF(NXT .EQ. PLUS) GO TO 1                                         GNLE    25
      IF(NXT .EQ. RPAR) GO TO 1                                         GNLE    26
      IF(NXT .EQ. MINUS) GO TO 1                                        GNLE    27
      IF(NXT .EQ. SLASH) GO TO 1                                        GNLE    28
      IF(NXT .EQ. COMMA) GO TO 1                                        GNLE    29
      IF(NXT .EQ. EQUALS) GO TO 1                                       GNLE    30
      GO TO 2                                                           GNLE    31
    1 JTYP=1                                                            GNLE    32
      M=1                                                               GNLE    33
      GO TO 90                                                          GNLE    34
    2 IF(NXT .NE. ASTRIK) GO TO 4                                       GNLE    35
      IF(NEXT(JPTR) .NE. ASTRIK) GO TO 1                                GNLE    36
      M=2                                                               GNLE    37
      JTYP=1                                                            GNLE    38
      GO TO 90                                                          GNLE    39
    4 IF(NXT .NE. LPAR) GO TO 40                                        GNLE    40
      IF(LSTART .EQ. 1) GO TO 10                                        GNLE    41
      IM1=LSTART-1                                                      GNLE    42
      IF(IPREV(IM1) .NE. 3) GO TO 1                                     GNLE    43
   10 CONTINUE                                                          GNLE    44
      NXT=NEXT(LSTART+1)                                                GNLE    45
      IF(NXT .EQ. BLANK) GO TO 120                                      GNLE    46
      IF(NXT .NE. PLUS .AND. NXT .NE. MINUS) GO TO 22                   GNLE    47
      IP=JPTR                                                           GNLE    48
      GO TO 24                                                          GNLE    49
   22 IP=JPTR-1                                                         GNLE    50
   24 CALL REALCK                                                       GNLE    51
      IF(IREAL .EQ. 0) GO TO 1                                          GNLE    52
      IF(IDES .EQ. 1) GO TO 1                                           GNLE    53
      IF(NEXT(IRELNO+1) .NE. COMMA) GO TO 1                             GNLE    54
      NXT=NEXT(JPTR)                                                    GNLE    55
      IF( NXT .NE. PLUS .AND. NXT .NE. MINUS) GO TO 30                  GNLE    56
```

```
      IP=JPTR                                                          GNLE      57
      GO TO 35                                                         GNLE      58
   30 IP=JPTR-1                                                        GNLE      59
   35 CALL REALCK                                                      GNLE      60
      IF(IREAL .EQ. 0) GO TO 120                                       GNLE      61
      IF(IDES .EQ. 1) GO TO 120                                        GNLE      62
      IF(NEXT(IRELND+1) .NE. RPAR) GO TO 120                           GNLE      63
      JTYP=6                                                           GNLE      64
      M=JPTR-LSTART                                                    GNLE      65
      GO TO 90                                                         GNLE      66
   40 IF(NXT .NE. DECPT) GO TO 50                                      GNLE      67
      ITP=ITYPE(JPTR)                                                  GNLE      68
      IF(ITP-2) 42,44,120                                              GNLE      69
   42 LOGST=LSTART+1                                                   GNLE      70
      CALL LOGCHK                                                      GNLE      71
      IF(LOG .EQ. 0) GO TO 120                                         GNLE      72
      JTYP=7                                                           GNLE      73
      M=JPTR-LSTART                                                    GNLE      74
      GO TO 90                                                         GNLE      75
   44 IP=LSTART                                                        GNLE      76
      CALL REALCK                                                      GNLE      77
      IF(IREAL .EQ. 0) GO TO 120                                       GNLE      78
      JTYP=4                                                           GNLE      79
      M=IRELND-LSTART+1                                                GNLE      80
      GO TO 90                                                         GNLE      81
   50 CONTINUE                                                         GNLE      82
      IF(ITYPE(LSTART) .NE. 2) GO TO 85                                GNLE      83
      IF(ITYP .EQ. 28) GO TO 54                                        GNLE      84
      IP=LSTART                                                        GNLE      85
      CALL REALCK                                                      GNLE      86
      IF(IREAL .EQ. 0) GO TO 54                                        GNLE      87
      JTYP=4                                                           GNLE      88
      M=IRELND-LSTART+1                                                GNLE      89
      GO TO 90                                                         GNLE      90
   54 JPTR=LSTART+1                                                    GNLE      91
   55 IF(ITYPE(JPTR) .EQ. 2) GO TO 57                                  GNLE      92
      GO TO 65                                                         GNLE      93
   57 IF(JPTR .GT. N) GO TO 60                                         GNLE      94
      GO TO 55                                                         GNLE      95
   60 M=N-LSTART+1                                                     GNLE      96
      JTYP=5                                                           GNLE      97
      GO TO 90                                                         GNLE      98
   65 IF(A(JPTR-1) .NE. AICH) GO TO 67                                 GNLE      99
      IF(ITYP .EQ. 8 .OR. ITYP .EQ. 28 .OR. ITYP .EQ. 27) GO TO 70     GNLE     100
   67 M=JPTR-LSTART-1                                                  GNLE     101
      JTYP=5                                                           GNLE     102
      GO TO 90                                                         GNLE     103
   70 IF(JPTR .GT. N) GO TO 120                                        GNLE     104
      M=N-LSTART+1                                                     GNLE     105
      IF(M .GT. 500) M=500                                             GNLE     106
      JTYP=3                                                           GNLE     107
      GO TO 90                                                         GNLE     108
   85 CONTINUE                                                         GNLE     109
      IF(ITYPE(LSTART) .NE. 1) GO TO 120                               GNLE     110
      IF(ITYP .EQ. 28) GO TO 100                                       GNLE     111
   88 CONTINUE                                                         GNLE     112
      IF(ITYPE(JPTR) .NE. 3) GO TO 86                                  GNLE     113
```

```
      M=JPTR-LSTART-1                                                    GNLE     114
      JTYP=2                                                             GNLE     115
      GO TO 90                                                           GNLE     116
   86 IF(JPTR .GT. N) GO TO 87                                           GNLE     117
      GO TO 88                                                           GNLE     118
  100 M=1                                                                GNLE     119
      JTYP=2                                                             GNLE     120
      GO TO 90                                                           GNLE     121
   87 M=N-LSTART+1                                                       GNLE     122
      JTYP=2                                                             GNLE     123
   90 CONTINUE                                                           GNLE     124
C** STORE THE NEXT ELEMENT IN "D"                                        GNLE     125
C** SQUEEZE BLANKS OUT OF STRING                                         GNLE     126
      DO 91 L=1,M                                                        GNLE     127
      LL=LSTART+L-1                                                      GNLE     128
      D(L)=A(LL)                                                         GNLE     129
   91 CONTINUE                                                           GNLE     130
      JPTR=LSTART+M                                                      GNLE     131
      CALL SQUEEZ                                                        GNLE     132
C** PROCESS THE ELEMENT                                                  GNLE     133
      CALL FORMEL                                                        GNLE     134
      RETURN                                                             GNLE     135
  120 CONTINUE                                                           GNLE     136
      JTYP=8                                                             GNLE     137
      RETURN                                                             GNLE     138
      END                                                                GNLE     139
```

```
      SUBROUTINE GOTO                                                    GOTO       2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),   RICH       2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,         CY58A     80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                   RICH       4
      COMMON/LABELS/STATRA(2,200),NLABEL                                 GOTO       4
      COMMON/BASBLK/IBLOCK(2500),NBLOCK,NB,NBRNCH                        CY58A     27
      DIMENSION IALPH(4)                                                 GOTO       6
      INTEGER STATRA,BLANK                                               GOTO       7
      INTEGER BITPUT                                                     GOTO       8
      DATA (IALPH(I),I=1,4)/1HG,1HO,1HT,1HO/                             GOTO       9
      DATA BLANK/1H /                                                    GOTO      10
C** "GO TO" STATEMENT PROCESSOR                                          GOTO      11
      DO 5 I=1,4                                                         GOTO      12
      IF(NEXT(JPTR) .NE. IALPH(I)) GO TO 10                              GOTO      13
    5 CONTINUE                                                           GOTO      14
C** GET STATEMENT LABEL                                                  GOTO      15
      CALL GNLE                                                          GOTO      16
      IF(JTYP .NE. 5) GO TO 10                                           GOTO      17
C** SEARCH STATEMENT NUMBER TABLE AND SET "GO TO" FLAG                   GOTO      18
      CALL STSRCH                                                        GOTO      19
      STATRA(2,LOC)=BITPUT(STATRA(2,LOC),1,12)                           GOTO      20
      IF(NEXT(JPTR) .NE. BLANK) GO TO 10                                 GOTO      21
C** STORE BRANCH IN BASIC BLOCK TABLE                                    GOTO      22
      NBLOCK=NBLOCK+1                                                    GOTO      23
      IBLOCK(NBLOCK)=LOC                                                 GOTO      24
      NBRNCH=1                                                           GOTO      25
      NB=1                                                               GOTO      26
      RETURN                                                             GOTO      27
   10 CALL ERROR(7)                                                      GOTO      28
      RETURN                                                             GOTO      29
      END                                                                GOTO      30
```

```
      SUBROUTINE GROUP                                                  GROUP      2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH       2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A    80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH       4
      COMMON/FORMAT/IDESST,IDESND,IGPST,IGPND,IGRP,SEPST,SEPND,         CY58A      3
     1 DIR,ICOM,ISEP                                                    GROUP      5
      INTEGER A,RPAR,SEPST,SEPND,DIR                                    GROUP      6
      DATA RPAR/1H)/                                                    GROUP      7
      IF(NEXT(IGPST+1) .EQ. RPAR) GO TO 20                              GROUP      8
C** THIS ROUTINE CHECKS THE SYNTAX OF A GROUP OF FIELD DESCRIPTORS      GROUP      9
C** AND RETURNS    IGRP=1 - VALID                                      GROUP     10
C**                IGRP=0 - INVALID                                    GROUP     11
      SEPST=JPTR-1                                                      GROUP     12
      DIR=1                                                             GROUP     13
      CALL SEPAR                                                        GROUP     14
C** CHECK INITIAL SEPARATOR                                            GROUP     15
      IF(ISEP .EQ. -1 .OR. ICOM .EQ. 1) GO TO 30                        GROUP     16
      IF(ISEP .EQ. 0) IDESST=SEPST                                      GROUP     17
      IF(ISEP .EQ. 1) IDESST=SEPND+1                                    GROUP     18
      IF(NEXT(IDESST) .EQ. RPAR) GO TO 20                               GROUP     19
C** CHECK FINAL SEPARATOR                                              GROUP     20
      SEPST=IGPND-1                                                     GROUP     21
      DIR=-1                                                            GROUP     22
      CALL SEPAR                                                        GROUP     23
      IF(ISEP .EQ. -1 .OR. ICOM .EQ. 1) GO TO 30                        GROUP     24
      DIR=1                                                             GROUP     25
   10 CONTINUE                                                          GROUP     26
C** CHECK NEXT DESCRIPTOR                                              GROUP     27
      CALL DESCRP                                                       GROUP     28
      IF(IDES .EQ. 0) GO TO 40                                          GROUP     29
      SEPST=IDESND+1                                                    GROUP     30
      IF(NEXT(SEPST) .EQ. RPAR) GO TO 20                                GROUP     31
C** CHECK NEXT SEPARATOR                                               GROUP     32
      CALL SEPAR                                                        GROUP     33
      IF(ISEP .EQ. 0 .OR. ISEP .EQ. -1) GO TO 30                        GROUP     34
      IDESST=SEPND+1                                                    GROUP     35
      IF(NEXT(IDESST) .NE. RPAR) GO TO 10                               GROUP     36
   20 IGRP=1                                                            GROUP     37
      RETURN                                                            GROUP     38
   30 IGRP=0                                                            GROUP     39
      CALL ERROR(7)                                                     GROUP     40
      RETURN                                                            GROUP     41
   40 CALL ERROR(88,IDESST)                                             GROUP     42
      IGRP=0                                                            GROUP     43
      RETURN                                                            GROUP     44
      END                                                               GROUP     45
```

```
      SUBROUTINE GRT                                                    GRT      2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH     2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,       CY58A    80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH     4
      COMMON/GLOBAL/NBLK,NREF,NSUBS,BLKTBL(200),EXTTBL(100),ISUBS(100)  GRT      4
      COMMON/LIST/NLIST,NINTFC,ISUBLT(2,200),INTFAC(300)                GRT      5
      INTEGER EXTTBL,BLKTBL,BLANK,BITPUT,BITGET                         GRT      6
      DATA BLANK/1H /                                                   GRT      7
C** THIS ROUTINE IS CALLED AFTER MODULE PROCESSING IS COMPLETE, TO       GRT      8
C** MAKE ENTRIES INTO THE GLOBAL REFERENCE TABLE                        GRT      9
      WRITE(6,1)                                                        GRT     10
    1 FORMAT(//)                                                        GRT     11
C** START WITH FIRST SUBPROGRAM NAME                                    GRT     12
      ISUB=INITID(2)                                                    GRT     13
      IF(ISUB .EQ. 0) GO TO 15                                          GRT     14
C** GET NEXT SUBPROGRAM NAME                                            GRT     15
   10 ISUB=IDTBL(2,ISUB)                                                GRT     16
      IF(ISUB .EQ. 0) GO TO 15                                          GRT     17
C** SKIP IF STATEMENT FUNCTION                                          GRT     18
      IF(BITGET(IDTBL(3,ISUB),19,1) .EQ. 1) GO TO 10                    GRT     19
      IF(NREF .EQ. 0) GO TO 4                                           GRT     20
C** SEARCH EXTERNAL REFERENCE TABLE TO SEE IF NAME ALREADY STORED        GRT     21
      DO 3 K=1,NREF                                                     GRT     22
      INDEX=EXTTBL(K)                                                   GRT     23
      IF(IDTBL(1,ISUB) .EQ. ISUBLT(1,INDEX)) GO TO 10                   GRT     24
    3 CONTINUE                                                          GRT     25
C** NAME NOT YET STORED - INCREMENT EXTERNAL REFERENCE COUNTER           GRT     26
    4 NREF=NREF+1                                                       GRT     27
      IF(NREF .GT. 100) GO TO 50                                        GRT     28
C** STORE SESCOMP LIST LOCATION OF NAME IN EXTERNAL REFERENCE TABLE      GRT     29
      EXTTBL(NREF)=BITGET(IDTBL(3,ISUB),36,9)                           GRT     30
      IF(MODE .EQ. 1) GO TO 10                                          GRT     31
C** ROLL CALL MODE - CHECK SUBPROGRAM CLASS                             GRT     32
      KLAS=BITGET(ISUBLT(2,EXTTBL(NREF)),10,4)                          GRT     33
C** IF SESCOMP MODULE - WRITE NAME ON AUXILIARY FILE FOR FURTHER USE     GRT     34
      IF(KLAS .EQ. 1 .OR. KLAS .EQ. 2) WRITE(9) IDTBL(1,ISUB)           GRT     35
      GO TO 10                                                          GRT     36
C** GET FIRST COMMON BLOCK NAME                                         GRT     37
   15 IBLK=INITID(3)                                                    GRT     38
   20 IF(IBLK .EQ. 0) RETURN                                            GRT     39
      IF(IDTBL(1,IBLK) .EQ. 1H ) GO TO 45                               GRT     40
C** SEARCH SYMBOL TABLE FOR NAME                                        GRT     41
      DO 25 I=1,NLIST                                                   GRT     42
      IF(IDTBL(1,IBLK) .NE. ISUBLT(1,I)) GO TO 25                       GRT     43
C** NAME FOUND IN SYMBOL TABLE                                          GRT     44
      LISTLC=I                                                          GRT     45
      IF(BITGET(ISUBLT(2,I),10,4) .EQ. 7) GO TO 30                      GRT     46
      IF(BITGET(ISUBLT(2,I),30,15) .NE. 0) GO TO 30                     GRT     47
C** NOT CATEGORY 1 COMMON BLOCK - GET SIZE AND STORE                    GRT     48
      ISZ=IDTBL(4,IBLK)                                                 GRT     49
      ISUBLT(2,I)=BITPUT(ISUBLT(2,I),ISZ,30)                            GRT     50
      GO TO 30                                                          GRT     51
   25 CONTINUE                                                          GRT     52
C** COMMON BLOCK NOT FOUND IN SESCOMP LIST - IF NOT BLANK COMMON,       GRT     53
C** ISSUE DIAGNOSTIC                                                    GRT     54
      CALL ERROR(62,IDTBL(1,IBLK))                                      GRT     55
C** STORE COMMON BLOCK NAME IN SESCOMP LIST                             GRT     56
```

```
      NLIST=NLIST+1                                                    GRT        57
      ISUBLT(1,NLIST)=IDTBL(1,IBLK)                                    GRT        58
C** STORE COMMON BLOCK SIZE IN SESCOMP LIST                            GRT        59
      ISZ=IDTBL(4,IBLK)                                                GRT        60
      ISUBLT(2,NLIST)=SHIFT(ISZ,30) .OR. SHIFT(10,50)                  GRT        61
      LISTLC=NLIST                                                     GRT        62
C** STORE LIST LOCATION IN SYMBOL TABLE                                GRT        63
   30 IDTBL(3,IBLK)=BITPUT(IDTBL(3,IBLK),LISTLC,36)                    GRT        64
      IF(NBLK .EQ. 0) GO TO 40                                         GRT        65
C** SEARCH COMMON BLOCK LIST FOR NAME                                  GRT        66
      DO 35 K=1,NBLK                                                   GRT        67
      IF(LISTLC .EQ. BLKTBL(K)) GO TO 45                               GRT        68
   35 CONTINUE                                                         GRT        69
C** NAME NOT FOUND IN COMMON BLOCK LIST                                GRT        70
C** INCREMENT COMMON BLOCK COUNTER AND STORE IN LIST                   GRT        71
   40 NBLK=NBLK+1                                                      GRT        72
      IF(NBLK .GT. 200) GO TO 60                                       GRT        73
      BLKTBL(NBLK)=LISTLC                                              GRT        74
   45 IBLK=IDTBL(2,IBLK)                                               GRT        75
      GO TO 20                                                         GRT        76
   50 CALL ERROR(59)                                                   GRT        77
      STOP                                                             GRT        78
   60 CALL ERROR(60)                                                   GRT        79
      STOP                                                             GRT        80
      END                                                              GRT        81
```

```
      SUBROUTINE IMPTYP                                                IMPTYP      2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH       2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A      80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                 RICH       4
      DIMENSION IALPH(6)                                               IMPTYP      4
      INTEGER D,BITPUT,BITGET                                          IMPTYP      5
      DATA (IALPH(I),I=1,6)/1HI,1HJ,1HK,1HL,1HM,1HN/                   IMPTYP      6
C** THIS ROUTINE CHECKS THAT THE VARIABLE TYPE HAS BEEN SET            IMPTYP      7
C** IF NOT, THE TYPE IS SET IMPLICITLY                                 IMPTYP      8
      IF(BITGET(IDTBL(3,LOC),11,1) .EQ. 1) GO TO 20                    IMPTYP      9
C** TYPE NOT YET SET, SET "TYPE SET" FLAG                              IMPTYP     10
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),1,11)                           IMPTYP     11
      DO 10 I=1,6                                                      IMPTYP     12
      IF(D(1) .NE. IALPH(I)) GO TO 10                                  IMPTYP     13
C** VARIABLE BEGINS WITH I,J,K,L,M, OR N - SET TYPE TO INTEGER         IMPTYP     14
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),4,10)                           IMPTYP     15
      GO TO 20                                                         IMPTYP     16
   10 CONTINUE                                                         IMPTYP     17
C** VARIABLE DOES NOT BEGIN WITH I,J,K,L,M, OR N - SET TYPE TO REAL    IMPTYP     18
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),1,10)                           IMPTYP     19
C** IF EXECUTABLE STATEMENT, SET FLAG                                  IMPTYP     20
   20 IF(ITYP .LE. 18) IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),1,38)          IMPTYP     21
      RETURN                                                           IMPTYP     22
      END                                                              IMPTYP     23
```

```
      SUBROUTINE INIT                                                    INIT      2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),   RICH      2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,         CY58A    80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                   RICH      4
      COMMON/FUNC/IFNCRA(5,12),MARGS,IARGS(50),FNCLOC(5),NFUNC           CY58A     7
      COMMON/STRING/NTYPE,NSTR,STR(500)                                  INIT      5
      COMMON/TYP/NQQ,RHSTYP,NQ2,NQ3,LHSTYP                              INIT      6
      COMMON/LIST/NLIST,NINTFC,ISUBLT(2,200),INTFAC(300)                 INIT      7
      COMMON/BASBLK/IBLOCK(2500),NBLOCK,NB,NBRNCH                        CY58A     8
      COMMON/STFUNC/NSTFNC,ISTFNC(10)                                    INIT      9
      INTEGER RHSTYP                                                     INIT     10
      INTEGER A,EQUALS,COMMA,RPAR,STR                                    INIT     11
      INTEGER BITPUT,BITGET,FNCLOC                                       INIT     12
      DATA EQUALS/1H=/,LPAR/1H(/,COMMA/1H,/,RPAR/1H)/                    INIT     13
C** ASSIGNMENT STATEMENT PROCESSOR                                       INIT     14
      IFN=0                                                              INIT     15
      NTYPE=1                                                            INIT     16
      IPTR=JPTR                                                          INIT     17
C** GET ASSIGNED VARIABLE                                                INIT     18
      CALL GNLE                                                          INIT     19
      IF(JTYP .NE. 2) GO TO 40                                           INIT     20
      CALL SEARCH                                                        INIT     21
      IF(NEXT(JPTR) .NE. EQUALS) GO TO 6                                 INIT     22
C** ASSIGNED VARIABLE IS NOT DIMENSIONED                                 INIT     23
      LOC2=0                                                             INIT     24
      IF(ISRCH(2) .NE. 1) GO TO 2                                        INIT     25
C** VARIABLE IS A FUNCTION, MUST BE THE FUNCTION NAME                    INIT     26
      IF(NXTID .NE. IFNCNM) CALL ERROR(10,NXTID)                         INIT     27
      IFN=1                                                              INIT     28
C** SET TYPE                                                             INIT     29
      CALL IMPTYP                                                        INIT     30
      LOC2=LOC                                                           INIT     31
    2 IF(ISRCH(1) .NE. 1) GO TO 18                                       INIT     32
      IF(LOC2 .EQ. 0) GO TO 4                                            INIT     33
      LOC=IDES                                                           INIT     34
      GO TO 5                                                            INIT     35
C** STORE IN SYMBOL TABLE                                                INIT     36
   18 IDTYP=1                                                            INIT     37
      CALL STORE                                                         INIT     38
      LOC=NID                                                            INIT     39
      IF(LOC2 .EQ. 0) GO TO 4                                            INIT     40
      IDTBL(3,LOC)=IDTBL(3,LOC2)                                         INIT     41
      GO TO 5                                                            INIT     42
C** SET TYPE                                                             INIT     43
    4 CALL IMPTYP                                                        INIT     44
C** SET "LEFT SIDE TYPE" TO VARIABLE TYPE                                INIT     45
    5 LHSTYP=BITGET(IDTBL(3,LOC),10,3)                                   INIT     46
      IF(LHSTYP .EQ. 5) NTYPE=2                                          INIT     47
      GO TO 30                                                           INIT     48
    6 IF(A(JPTR-1) .NE. LPAR) GO TO 40                                   INIT     49
      IF(ISRCH(1) .EQ. 1) GO TO 12                                       INIT     50
      IF(ISRCH(2) .NE. 1) GO TO 15                                       INIT     51
      CALL ERROR(10,NXTID)                                               INIT     52
      GO TO 6                                                            INIT     53
C** FUNCTION DEFINING STATEMENT                                          INIT     54
C** STORE VARIABLE AS FUNCTION                                           INIT     55
   15 IDTYP=2                                                            INIT     56
```

```
      CALL STORE                                                        INIT      57
      LOC=NID                                                           INIT      58
      GO TO 8                                                           INIT      59
    7 CALL SWITCH                                                       INIT      60
C** SET TYPE                                                            INIT      61
    8 CALL IMPTYP                                                       INIT      62
      LHSTYP=BITGET(IDTBL(3,LOC),10,3)                                  INIT      63
      IF(LHSTYP .EQ. 5) NTYPE=2                                         INIT      64
      NARG=0                                                            INIT      65
      ITYP=35                                                           INIT      66
C** STORE IN STATEMENT FUNCTION TABLE                                   INIT      67
      NSTFNC=NSTFNC+1                                                   INIT      68
      IF(NSTFNC .GT. 10) GO TO 60                                       CY58A      9
      ISTFNC(NSTFNC)=LOC                                                INIT      69
      LOC1=LOC                                                          INIT      70
C** SET "STATEMENT FUNCTION" FLAG                                       INIT      71
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),1,19)                            INIT      72
C** GET NEXT ARGUMENT                                                   INIT      73
   10 CALL GNLE                                                         INIT      74
      IF(JTYP .NE. 2) GO TO 50                                          INIT      75
C** STORE ARGUMENT IN SYMBOL TABLE                                      INIT      76
      NARG=NARG+1                                                       INIT      77
      CALL SEARCH                                                       INIT      78
      IF(ISRCH(2) .EQ. 1) CALL ERROR(54,NARG)                           INIT      79
      IF(ISRCH(1) .EQ. 1) GO TO 20                                      INIT      80
      IDTYP=1                                                           INIT      81
      CALL STORE                                                        INIT      82
      LOC=NID                                                           INIT      83
      GO TO 25                                                          INIT      84
   20 IF(BITGET(IDTBL(3,LOC),1,1) .EQ. 1) CALL ERROR(54,NARG)           INIT      85
   25 CALL IMPTYP                                                       INIT      86
      IF(NEXT(JPTR) .EQ. COMMA) GO TO 10                                INIT      87
      IF(A(JPTR-1) .NE. RPAR) GO TO 40                                  INIT      88
C** STORE NO. OF ARGUMENTS IN SYMBOL TABLE                              INIT      89
      IDTBL(3,LOC1)=BITPUT(IDTBL(3,LOC1),NARG,7)                        INIT      90
      IF(NEXT(JPTR) .EQ. EQUALS) GO TO 32                               INIT      91
      GO TO 40                                                          INIT      92
   12 IF(BITGET(IDTBL(3,LOC),1,1) .NE. 1) GO TO 7                       INIT      93
C** VARIABLE IS DIMENSIONED                                             INIT      94
C** SET TYPE                                                            INIT      95
      CALL IMPTYP                                                       INIT      96
      LHSTYP=BITGET(IDTBL(3,LOC),10,3)                                  INIT      97
      IF(LHSTYP .EQ. 5) NTYPE=2                                         INIT      98
      JPTR=IPTR                                                         INIT      99
      JBLOCK=NBLOCK+1                                                   INIT     100
C** PARSE THE LEFT-HAND SIDE                                            INIT     101
      CALL EXPR                                                         INIT     102
      CALL PARSE                                                        INIT     103
C** STORE BASIC BLOCKS                                                  INIT     104
      CALL BLKSTR                                                       INIT     105
      IBLOCK(JBLOCK)=IBLOCK(JBLOCK) - 1000                              INIT     106
      GO TO 32                                                          INIT     107
C** SIMPLE VARIABLE, STORE IN BASIC BLOCK TABLE                         INIT     108
   30 NBLOCK=NBLOCK+1                                                   INIT     109
      JBLOCK=NBLOCK                                                     INIT     110
      IBLOCK(NBLOCK)=1000+LOC                                           INIT     111
   32 NTMS=0                                                            INIT     112
```

```
      IPTR=JPTR                                                          INIT     113
C** PARSE THE RIGHT-HAND SIDE                                            INIT     114
      CALL EXPR                                                          INIT     115
      IF(JPTR .LE. N) CALL ERROR(7)                                      INIT     116
      CALL PARSE                                                         INIT     117
C** PROCESS FUNCTION REFERENCES                                          INIT     118
      CALL FNCSTR                                                        INIT     119
C** CHECK IF ASSIGNMENT IS LEGAL                                         INIT     120
      CALL EXPRCK                                                        INIT     121
      IF(ITYP .EQ. 35) GO TO 36                                          INIT     122
C** STORE BASIC BLOCKS FOR RIGHT-HAND SIDE                               INIT     123
      CALL BLKSTR                                                        INIT     124
      IBLOCK(NBLOCK+1)=IBLOCK(JBLOCK)                                    INIT     125
      DO 34 K=JBLOCK,NBLOCK                                              INIT     126
   34 IBLOCK(K)=IBLOCK(K+1)                                              INIT     127
      IF(LTYP .EQ. 1) RETURN                                             INIT     128
   36 IF(MODE .NE. 1) GO TO 35                                           INIT     129
      IF(RHSTYP .EQ. 3 .OR. RHSTYP .EQ. 4) RETURN                        INIT     130
      IF(NSTR .LT. 6) RETURN                                             INIT     131
      JPTR=IPTR-1                                                        INIT     132
C** VARIABLE PRECISION MODE                                              INIT     133
C** GENERATE CALLS TO VARIABLE PRECISION SUBROUTINES                      INIT     134
      CALL CNVRT                                                         INIT     135
      RETURN                                                             INIT     136
C** ROLL CALL MODE                                                       INIT     137
C** GENERATE CALL TO "ROLCHK" IF NECESSARY                                INIT     138
   35 IF(NFUNC .EQ. 0) RETURN                                            INIT     139
      IF(IFN .EQ. 1) RETURN                                              INIT     140
C** LOOK AT EVERY FUNCTION CALL IN STATEMENT                             INIT     141
      DO 38 J=1,NFUNC                                                    INIT     142
      LOC=FNCLOC(J)                                                      INIT     143
      INDEX=BITGET(IDTBL(3,LOC),36,9)                                    INIT     144
      KLAS=BITGET(ISUBLT(2,INDEX),10,4)                                  INIT     145
      IF(KLAS .NE. 1 .AND. KLAS .NE. 2) GO TO 38                         INIT     146
C** SUBROUTINE OR FUNCTION MODULE - ISSUE A CALL TO "ROLCHK"              INIT     147
      CALL CALL2                                                         INIT     148
      RETURN                                                             INIT     149
   38 CONTINUE                                                           INIT     150
      RETURN                                                             INIT     151
   40 CALL ERROR(7)                                                      INIT     152
      RETURN                                                             INIT     153
   50 CALL ERROR(15)                                                     INIT     154
      RETURN                                                             INIT     155
   60 CALL ERROR(89)                                                     CY58A     10
      RETURN                                                             CY58A     11
      END                                                                INIT     156
```

```
      SUBROUTINE INTRIN                                                  INTRIN     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),   RICH      2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,         CY58A    80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                   RICH      4
      COMMON/LIST/NLIST,NINTFC,ISUBLT(2,200),INTFAC(300)                 INTRIN     4
      INTEGER BITGET                                                     INTRIN     5
C** THIS ROUTINE IS CALLED AFTER PROCESSING THE MODULE, TO CHECK THAT     INTRIN     6
C** NO INTRINSIC FUNCTION NAMES HAVE BEEN MISUSED                        INTRIN     7
      DO 100 I=1,NLIST                                                   INTRIN     8
      IF(BITGET(ISUBLT(2,I),10,4) .NE. 4) GO TO 100                      INTRIN     9
C** GET NEXT INTRINSIC FUNCTION NAME FROM SESCOMP LIST                   INTRIN    10
      NXTID=ISUBLT(1,I)                                                  INTRIN    11
C** SEARCH SYMBOL TABLE FOR NAME                                         INTRIN    12
      CALL SEARCH                                                        INTRIN    13
      CALL COMSCH                                                        INTRIN    14
C** IF FOUND, ISSUE DIAGNOSTIC                                           INTRIN    15
      IF(ISRCH(1) .EQ. 1 .OR. ISRCH(3) .EQ. 1) CALL ERROR(74,NXTID)      INTRIN    16
  100 CONTINUE                                                           INTRIN    17
      RETURN                                                             INTRIN    18
      END                                                                INTRIN    19
```

```
      SUBROUTINE IO                                                    IO        2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH      2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,       CY58A    88
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                 RICH      4
      COMMON/STRING/NTYPE,NSTR,STR(500)                                IO        4
      COMMON/LABELS/STATRA(2,200),NLABEL                               IO        5
      COMMON/BASBLK/IBLOCK(2500),NBLOCK,NB,NBRNCH                      CY58A    33
      COMMON/INPOUT/NCALL,IN,IOP                                       IO        7
      DIMENSION IALPH1(4),IALPH2(5),IALPH3(8)                          IO        8
      INTEGER A,STATRA,RPAR,COMMA,BLANK                                IO        9
      INTEGER BITPUT,BITGET                                            IO       10
      DATA LPAR/1H(/,RPAR/1H)/,COMMA/1H,/,BLANK/1H /                   IO       11
      DATA (IALPH1(I),I=1,4)/1HR,1HE,1HA,1HD/                          IO       12
      DATA (IALPH2(I),I=1,5)/1HW,1HR,1HI,1HT,1HE/                      IO       13
      DATA (IALPH3(I),I=1,8)/1HC,1HO,1HN,1HT,1HI,1HN,1HU,1HE/          IO       14
C** I/O STATEMENT PROCESSOR                                            IO       15
      IFRMT=0                                                          IO       16
      IF(ITYP .EQ. 12) GO TO 10                                        IO       17
      DO 5 I=1,4                                                       IO       18
      IF(NEXT(JPTR) .NE. IALPH1(I)) GO TO 50                           IO       19
    5 CONTINUE                                                         IO       20
      GO TO 20                                                         IO       21
C** WRITE STATEMENT                                                    IO       22
   10 DO 15 I=1,5                                                      IO       23
      IF(NEXT(JPTR) .NE. IALPH2(I)) GO TO 50                           IO       24
   15 CONTINUE                                                         IO       25
   20 IF(NEXT(JPTR) .NE. LPAR) GO TO 50                                IO       26
C** GET I/O DEVICE                                                     IO       27
      CALL GNLE                                                        IO       28
C** IF NOT VARIABLE, ISSUE DIAGNOSTIC                                  IO       29
      IF(JTYP .EQ. 2) GO TO 22                                         IO       30
C** READ STATEMENT                                                     IO       31
      CALL ERROR(22)                                                   IO       32
      GO TO 28                                                         IO       33
C** VARIABLE I/O DEVICE - GET SYMBOL TABLE LOCATION                    IO       34
   22 CALL SEARCH                                                      IO       35
      IF(ISRCH(2) .EQ. 1) CALL ERROR(10,NXTID)                         IO       36
      IF(ISRCH(1) .EQ. 1) GO TO 25                                     IO       37
      IDTYP =1                                                         IO       38
      CALL STORE                                                       IO       39
      LOC=NID                                                          IO       40
C** CHECK THAT THE DEVICE IS INTEGER VARIABLE                          IO       41
   25 CALL IMPTYP                                                      IO       42
      IF(BITGET(IDTBL(3,LOC),1,1) .EQ. 1) CALL ERROR(14,NXTID)         IO       43
      IF(BITGET(IDTBL(3,LOC),10,3) .NE. 4) CALL ERROR(22)              IO       44
C** STORE IN BASIC BLOCK TABLE                                         IO       45
      NBLOCK=NBLOCK+1                                                  IO       46
      IBLOCK(NBLOCK)=2000+LOC                                          IO       47
   28 IF(NEXT(JPTR) .EQ. COMMA) GO TO 40                               IO       48
      JPTR=JPTR-1                                                      IO       49
      GO TO 30                                                         IO       50
C** FORMATTED I/O STATEMENT - GET STATEMENT NUMBER                     IO       51
   40 CALL GNLE                                                        IO       52
      IF(JTYP .NE. 5) GO TO 26                                         IO       53
C** GET STATEMENT NUMBER TABLE LOCATION AND SET "REFERENCED" FLAG      IO       54
      CALL STSRCH                                                      IO       55
      STATRA(2,LOC)=BITPUT(STATRA(2,LOC),1,12)                         IO       56
```

```
      GO TO 29                                                          IO         57
   26 IF(JTYP .NE. 2) GO TO 50                                          IO         58
C** VARIABLE FORMAT - GET SYMBOL TABLE LOCATION                         IO         59
      CALL SEARCH                                                       IO         60
      IF(ISRCH(2) .EQ. 1) CALL ERROR(10,NXTID)                          IO         61
      IF(ISRCH(1) .EQ. 1) GO TO 27                                      IO         62
      IDTYP=1                                                           IO         63
      CALL STORE                                                        IO         64
      LOC=NID                                                           IO         65
   27 CALL IMPTYP                                                       IO         66
C** CHECK THAT VARIABLE FORMAT IS AN ARRAY                              IO         67
      IF(BITGET(IDTBL(3,LOC),1,1) .NE. 1) CALL ERROR(43)                IO         68
   29 IFRMT=1                                                           IO         69
   30 IF(NEXT(JPTR) .NE. RPAR) GO TO 50                                 IO         70
      IF(NEXT(JPTR) .NE. BLANK) GO TO 35                                IO         71
C** NO I/O LIST - MUST NOT BE UNFORMATTED WRITE                         IO         72
      IF(ITYP .EQ. 12 .AND. IFRMT .EQ. 0) CALL ERROR(44)                IO         73
      GO TO 36                                                          IO         74
C** STATEMENT HAS AN I/O LIST                                           IO         75
   35 JPTR=JPTR-1                                                       IO         76
C** PARSE THE I/O LIST                                                  IO         77
      CALL EXPR                                                         IO         78
      NTYPE=3                                                           IO         79
      CALL PARSE                                                        IO         80
C** STORE BASIC BLOCKS                                                  IO         81
      CALL IOSTR                                                        IO         82
   36 IF(MODE .NE. 1 .AND. ITYP .EQ. 11) GO TO 37                       IO         83
      RETURN                                                            IO         84
C** READ STATEMENT IN ROLL CALL MODE                                    IO         85
   37 IF(ITYPE(1) .EQ. 2) GO TO 42                                      IO         86
C** MAKE READ STATEMENT A COMMENT STATEMENT                             IO         87
      A(1)=1HC                                                          IO         88
      RETURN                                                            IO         89
C** STATEMENT HAS LABEL - GENERATE CONTINUE STATEMENT WITH SAME LABEL    IO         90
   42 WRITE(IOP,45) (A(I),I=1,6),(IALPH3(I),I=1,8)                      IO         91
   45 FORMAT(72A1)                                                      IO         92
C** MAKE READ STATEMENT A COMMENT AND DELETE LABEL                      IO         93
      A(1)=1HC                                                          IO         94
      DO 47 I=2,6                                                       IO         95
   47 A(I)=1H                                                           IO         96
      RETURN                                                            IO         97
   50 CALL ERROR(7)                                                     IO         98
      RETURN                                                            IO         99
      END                                                               IO        100
```

```
      SUBROUTINE IOSTR                                                  IOSTR      2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH       2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A     80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH       4
      COMMON/FUNC/IFNCRA(5,12),MARGS,IARGS(50),FNCLOC(5),NFUNC          CY58A     34
      COMMON/BASBLK/IBLOCK(2500),NBLOCK,N8,NBRNCH                       CY58A     35
      INTEGER BITGET                                                    IOSTR      6
C** THIS ROUTINE IS CALLED AFTER PARSING AN I/O LIST, TO STORE          IOSTR      7
C** INFORMATION IN THE BASIC BLOCK TABLE                               IOSTR      8
      DO 100 I=1,MARGS                                                  IOSTR      9
C** INCREMENT BLOCK COUNTER                                            IOSTR     10
      NBLOCK=NBLOCK+1                                                   IOSTR     11
      ICOL=20*MOD(I-1,3)+10                                             IOSTR     12
      IVR=(I+2)/3                                                       IOSTR     13
C** GET SYMBOL TABLE LOCATION OF VARIABLE                              IOSTR     14
      LOC=BITGET(IARGS(IVR),ICOL,10)                                    IOSTR     15
C** GET CLASS OF VARIABLE     ISUB=0  -  I/O VARIABLE                 IOSTR     16
C**                           ISUB=1  -  SUBSCRIPT                    IOSTR     17
C**                           ISUB=2  -  IMPLIED DO INDEX             IOSTR     18
      ISUB=BITGET(IARGS(IVR),ICOL+5,2)                                  IOSTR     19
      IF(ISUB .EQ. 1) GO TO 90                                          IOSTR     20
      IF(ISUB .EQ. 2) GO TO 20                                          IOSTR     21
      IF(ITYP .EQ. 11) GO TO 80                                         IOSTR     22
      IF(ITYP .EQ. 12) GO TO 90                                         IOSTR     23
   20 IDEF=BITGET(IARGS(IVR),ICOL+10,5)                                 IOSTR     24
C** IMPLIED DO                                                         IOSTR     25
      NMOVE=I-IDEF-1                                                    IOSTR     26
C** MOVE DO INDEX TO BEGINNING OF BLOCK                                IOSTR     27
      DO 30 J=1,NMOVE                                                   IOSTR     28
      ITEMP=NBLOCK-J                                                    IOSTR     29
   30 IBLOCK(ITEMP+1)=IBLOCK(ITEMP)                                     IOSTR     30
      IBLOCK(ITEMP)=1000+LOC                                            IOSTR     31
C** UNDEFINE DO INDEX AT END OF BLOCK                                  IOSTR     32
      NBLOCK=NBLOCK+1                                                   IOSTR     33
      IBLOCK(NBLOCK)=6000+LOC                                           IOSTR     34
      GO TO 100                                                         IOSTR     35
C** VARIABLE IS DEFINED - STORE IN BASIC BLOCK TABLE                   IOSTR     36
   80 IBLOCK(NBLOCK)=1000+LOC                                           IOSTR     37
      GO TO 100                                                         IOSTR     38
C** VARIABLE IS REFERENCED - STORE IN BASIC BLOCK TABLE                 IOSTR     39
   90 IBLOCK(NBLOCK)=2000+LOC                                           IOSTR     40
  100 CONTINUE                                                          IOSTR     41
      RETURN                                                            IOSTR     42
      END                                                               IOSTR     43
```

```
      FUNCTION IPREV(IA)                                                IPREV     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH      2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A    80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IOES                                  RICH      4
      INTEGER A,BLANK                                                   IPREV     4
      DATA BLANK/1H /                                                   IPREV     5
C** THIS FUNCTION CALLS "ITYPE" TO GET THE CLASS OF THE                 IPREV     6
C** PREVIOUS CHARACTER                                                  IPREV     7
      DO 10 I=1,N                                                       IPREV     8
      J=IA-I+1                                                          IPREV     9
      IF(J .EQ. 0) GO TO 20                                             IPREV    10
      IF(A(J) .EQ. BLANK) GO TO 10                                      IPREV    11
      IPREV=ITYPE(J)                                                    IPREV    12
      JPTR=J-1                                                          IPREV    13
      RETURN                                                            IPREV    14
   10 CONTINUE                                                          IPREV    15
   20 IPREV=3                                                           IPREV    16
      JPTR=0                                                            IPREV    17
      RETURN                                                            IPREV    18
      END                                                               IPREV    19
```

```
      FUNCTION ITYPE(IO)                                                ITYPE     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH      2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A    80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IOES                                  RICH      4
      INTEGER BITGET                                                    ITYPE     4
C** THIS FUNCTION CALLS "NEXT" TO GET THE NEXT CHARACTER IN THE INPUT   ITYPE     5
C** STRING AND RETURNS        ITYPE=1 - LETTER                         ITYPE     6
C**                           ITYPE=2 - DIGIT                          ITYPE     7
C**                           ITYPE=3 - OTHER                          ITYPE     8
                                                                        ITYPE     9
      NXT=NEXT(IO)                                                      ITYPE    10
      IVAL=BITGET(NXT,6,6)                                              ITYPE    11
C****IF UNIVAC 1108 - PLACE A "C" IN COLUMN 1 OF NEXT TWO CARDS         ITYPE    12
      IF(IVAL .GE. 1RA .AND. IVAL .LE. 1RZ) GO TO 10                    ITYPE    13
      IF(IVAL .GE. 1R0 .AND. IVAL .LE. 1R9) GO TO 20                    ITYPE    14
C****IF CDC 6700 - PLACE A "C" IN COLUMN 1 OF NEXT TWO CARDS            ITYPE    15
C     IF(IVAL .GE. 06 .AND. IVAL .LE. 037) GO TO 10                     ITYPE    16
C     IF(IVAL .GE. 060 .AND. IVAL .LE. 071) GO TO 20                    ITYPE    17
      ITYPE=3                                                           ITYPE    18
      IF(IVAL .GT. 1R.) CALL ERROR(23)                                  ITYPE    19
      RETURN                                                            ITYPE    20
   10 ITYPE=1                                                           ITYPE    21
      RETURN                                                            ITYPE    22
   20 ITYPE=2                                                           ITYPE    23
      RETURN                                                            ITYPE    24
      END
```

```
      SUBROUTINE LOGCHK                                                 LOGCHK     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH       2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A     80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH       4
      COMMON/LOGIC/LOG,LOGST                                            LOGCHK     4
      DIMENSION LOGOP(11),LOGRA(5)                                      LOGCHK     5
      DATA (LOGOP(I),I=1,11)/2HLT,2HLE,2HGT,2HGE,2HEQ,2HNE,2HOR,3HAND, LOGCHK     6
     * 3HNOT,4HTRUE,5HFALSE/                                            LOGCHK     7
      JPTR=LOGST                                                        LOGCHK     8
C** FORM CHARACTER STRING CONTAINING LOGICAL OPERATOR                   LOGCHK     9
      DO 10 I=1,6                                                       LOGCHK    10
      NXT=NEXT(JPTR)                                                    LOGCHK    11
      IF(NXT .EQ. 1H.) GO TO 12                                         LOGCHK    12
   10 LOGRA(I)=NXT                                                      LOGCHK    13
      GO TO 20                                                          LOGCHK    14
   12 IF(I .LT. 3) GO TO 20                                             LOGCHK    15
C** PACK THE LOGICAL OPERATOR                                           LOGCHK    16
      CALL CAA(LOGRA,I-1,LOG)                                           LOGCHK    17
C** COMPARE WITH VALID OPERATORS                                        LOGCHK    18
      DO 15 I=1,11                                                      LOGCHK    19
      IF(LOG .EQ. LOGOP(I)) GO TO 30                                    LOGCHK    20
C** THIS ROUTINE CHECKS TO SEE IF A LOGICAL OPERATOR OR CONSTANT IS VALI LOGCHK    21
   15 CONTINUE                                                          LOGCHK    22
   20 LOG=0                                                             LOGCHK    23
C** INVALID OPERATOR                                                    LOGCHK    24
      RETURN                                                            LOGCHK    25
   30 LOG=1                                                             LOGCHK    26
C** VALID OPERATOR                                                      LOGCHK    27
      LOGID=I                                                           LOGCHK    28
      RETURN                                                            LOGCHK    29
      END                                                               LOGCHK    30
```

```
      SUBROUTINE LOGIF                                                  LOGIF     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH      2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A    80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IOES                                  RICH      4
      COMMON/STRING/NTYPE,NSTR,STR(500)                                 LOGIF     4
      COMMON/BASBLK/IBLOCK(2500),NBLOCK,NB,NBRNCH                       CY58A    36
      INTEGER A,STR,BLANK,AY,EF                                         LOGIF     6
      DATA LPAR/1H(/,BLANK/1H /,AY/1HI/,EF/1HF/                         LOGIF     7
C** "LOGICAL IF" STATEMENT PROCESSOR                                    LOGIF     8
      IF(NEXT(JPTR) .NE. AY) GO TO 110                                  LOGIF     9
      IF(NEXT(JPTR) .NE. EF) GO TO 110                                  LOGIF    10
      IF(NEXT(JPTR) .NE. LPAR) GO TO 110                                LOGIF    11
      JPTR=JPTR-1                                                       LOGIF    12
C** PARSE THE LOGICAL EXPRESSION                                        LOGIF    13
      CALL EXPR                                                         LOGIF    14
      NSTR=NSTR+1                                                       LOGIF    15
      STR(NSTR)= -5                                                     LOGIF    16
      NTYPE=2                                                           LOGIF    17
      CALL PARSE                                                        LOGIF    18
C** PROCESS FUNCTION REFERENCES                                         LOGIF    19
      CALL FNCSTR                                                       LOGIF    20
C** STORE BASIC BLOCKS                                                  LOGIF    21
      CALL BLKSTR                                                       LOGIF    22
      IF(ITYP .GT. 15) GO TO 130                                        LOGIF    23
      LTYP=1                                                            LOGIF    24
C** PROCESS STATEMENT FOLLOWING LOGICAL EXPRESSION BY GOING TO THE      LOGIF    25
C** APPROPRIATE STATEMENT PROCESSOR                                     LOGIF    26
      GO TO (10,20,30,40,50,60,70,80,70,70,90,90,100,100,100),ITYP      LOGIF    27
   10 CALL INIT                                                         LOGIF    28
      RETURN                                                            LOGIF    29
   20 CALL ASSIGN                                                       LOGIF    30
      RETURN                                                            LOGIF    31
   30 CALL GOTO                                                         LOGIF    32
   35 NBLOCK=NBLOCK+1                                                   LOGIF    33
      IBLOCK(NBLOCK)=998                                                LOGIF    34
      NBRNCH=2                                                          LOGIF    35
      RETURN                                                            LOGIF    36
   40 CALL ASGOTO                                                       LOGIF    37
   45 NBLOCK=NBLOCK+1                                                   LOGIF    38
      IBLOCK(NBLOCK)=998                                                LOGIF    39
      NBRNCH=NBRNCH+1                                                   LOGIF    40
      RETURN                                                            LOGIF    41
   50 CALL CTGOTO                                                       LOGIF    42
      GO TO 45                                                          LOGIF    43
   60 CALL ARIF                                                         LOGIF    44
      GO TO 45                                                          LOGIF    45
   70 CALL SIMP                                                         LOGIF    46
      IF(ITYP .EQ. 7) RETURN                                            LOGIF    47
      GO TO 35                                                          LOGIF    48
   80 CALL CALL                                                         LOGIF    49
      RETURN                                                            LOGIF    50
   90 CALL IO                                                           LOGIF    51
      RETURN                                                            LOGIF    52
  100 CALL AUXIO                                                        LOGIF    53
      RETURN                                                            LOGIF    54
  110 CALL ERROR(7)                                                     LOGIF    55
      RETURN                                                            LOGIF    56
  130 CALL ERROR(45)                                                    LOGIF    57
      RETURN                                                            LOGIF    58
      END                                                               LOGIF    59
```

```
      SUBROUTINE LOOPCK                                                 LOOPCK     2
      COMMON/LABELS/STATRA(2,200),NLABEL                                LOOPCK     3
      COMMON/DOLOOP/ISTACK(4,50),NSTACK,ILOOP                           LOOPCK     4
      COMMON/BASBLK/IBLOCK(2500),NBLOCK,NB,NBRNCH                       CY58A     49
      INTEGER STATRA,BITGET                                             LOOPCK     6
C** THIS ROUTINE IS CALLED AFTER MODULE PROCESSING TO CHECK DO LOOP      LOOPCK     7
C** STRUCTURE                                                           LOOPCK     8
      IF(NSTACK .EQ. 0) RETURN                                          LOOPCK     9
C** START AT BEGINNING OF BASIC BLOCK TABLE                            LOOPCK    10
      IBLKST=1                                                          LOOPCK    11
C** GET LAST LOCATION IN BLOCK                                         LOOPCK    12
   10 IBLKND=BITGET(IBLOCK(IBLKST),28,16)-1                             LOOPCK    13
C** GET LOOP WHICH CONTAINS BLOCK                                      LOOPCK    14
      LOOP2=BITGET(IBLOCK(IBLKST),12,6)                                 LOOPCK    15
C** GET NO. OF BRANCHES FROM BLOCK                                     LOOPCK    16
      NBR=BITGET(IBLOCK(IBLKST),6,6)                                    LOOPCK    17
      IF(IBLKND .EQ. -1) IBLKND=NBLOCK                                  LOOPCK    18
      IST=IBLKND-NBR+1                                                  LOOPCK    19
C** THIS LOOP EXAMINES ALL BRANCHES FROM THE BLOCK                     LOOPCK    20
C** AND CHECKS THAT THEY ARE VALID                                     LOOPCK    21
      DO 100 I=IST,IBLKND                                               LOOPCK    22
      JLOOP=LOOP2                                                       LOOPCK    23
      IF(IBLOCK(I) .GE. 998) GO TO 100                                  LOOPCK    24
      IBLK=IBLOCK(I)                                                    LOOPCK    25
C** GET BASIC BLOCK WHICH CONTAINS BRANCH                              LOOPCK    26
      NXTBLK=BITGET(STATRA(2,IBLK),36,18)                               LOOPCK    27
C** GET DO LOOP WHICH CONTAINS THE BLOCK                               LOOPCK    28
      KLOOP=BITGET(IBLOCK(NXTBLK),12,6)                                 LOOPCK    29
      IF(KLOOP .EQ. 0) GO TO 100                                        LOOPCK    30
      IF(JLOOP .EQ. 0) GO TO 200                                        LOOPCK    31
C** BOTH BLOCKS ARE CONTAINED IN DO LOOPS                              LOOPCK    32
C** TRAVERSE THE DO STACK TO DETERMINE IF BRANCH IS LEGAL               LOOPCK    33
   50 IF(JLOOP .EQ. KLOOP) GO TO 100                                    LOOPCK    34
      JLOOP=ISTACK(3,JLOOP)                                             LOOPCK    35
      IF(JLOOP .EQ. 0) GO TO 200                                        LOOPCK    36
      GO TO 50                                                          LOOPCK    37
  200 IBLK=IBLOCK(I)                                                    LOOPCK    38
      WRITE(6,201) STATRA(1,IBLK)                                       LOOPCK    39
  201 FORMAT(6X,65H ILLEGAL TRANSFER INTO THE RANGE OF A DO LOOP AT STAT LOOPCK    40
     1EMENT NUMBER,I6)                                                  LOOPCK    41
  100 CONTINUE                                                          LOOPCK    42
      IF(IBLKND .EQ. NBLOCK) RETURN                                     LOOPCK    43
C** GET START OF NEXT BLOCK                                            LOOPCK    44
      IBLKST=IBLKND+1                                                   LOOPCK    45
      GO TO 10                                                          LOOPCK    46
      END                                                               LOOPCK    47
```

```
      SUBROUTINE LVDLET                                                 LVDLET 2
      COMMON/LVARGS/IFUNC,IARG,IADD,IPOS,ITYP,IVAL,LSTHED,NVAL,         LVDLET 3
     +  IDSTRY,IVALS(10),ITYP1(10),NSKIP                               LVDLET 4
      INTEGER FLGSPC,FL0MSK,FL1MSK,FL2MSK,FL5MSK,FLG67,REGASP,THIS      LVDLET 5
     + ,FL3MSK,FL4MSK,SEQSPC                                           LVDLET 6
      COMMON/LVVTR1/MEMSZE,REGASP,NODSPC(    1)/LVVTR2/LSTSPC(    1)/ LVDLET 7
     *LVVTR3/LNKSPC(    1)/LVVTR4/FLGSPC(    1)                       LVDLET 8
      COMMON/LVFLAG/FL0MSK,FL1MSK,FL2MSK,FL3MSK,FL4MSK,FL5MSK,FLG67     LVDLET 9
      COMMON /LVTABL/ MAPSZE,MAP(1) /LVVSEQ/ ISEQSZ,SEQSPC(1)           LVDLET10
      DATA NFLG02/1378/                                                 LVDLET11
      KONFLC=0                                                          LVDLET12
C     DETERMINE DIRECTION TO PROCEED FOR MULTIVALUE LISTS               LVDLET13
      JPOS=IPOS                                                         LVDLET14
      IPOS=IABS(IPOS)                                                   LVDLET15
      IF(IADD.NE.-1)  GO TO  74                                         LVDLET16
      IF(IARG.EQ.-1) GO TO 66                                           LVDLET17
      IADD=IFUNC+IARG                                                   LVDLET18
      IF(IADD.GT.MEMSZE) IADD=IADD-MEMSZE                               LVDLET19
      IF((FLGSPC(IADD).AND.FL5MSK).EQ.0)     GO TO 99                   LVDLET20
  1   IF(NODSPC(IADD).EQ.IARG) GO TO 4                                  LVDLET21
C     SEARCH CONFLICT LIST FOR THE FUNCTION                             LVDLET22
      IADD=LNKSPC(IADD)                                                 LVDLET23
      IF((FLGSPC(IADD).AND. FL5MSK) .NE.0) GO TO 99                     LVDLET24
      GO TO 1                                                           LVDLET25
 66   IADD=IFUNC                                                        LVDLET26
C     TO DELETE A SPECIFIC TYPE OF NODE (INDEXED DELETE), GO TO 72      LVDLET27
  4   IF(ITYP.NE.-1) GO TO 72                                           LVDLET28
      IF((FLGSPC(IADD).AND.FL0MSK).EQ.0)  GO TO 6                       LVDLET29
      ISADD=LSTSPC(IADD)                                                LVDLET30
C     DELETE ENTIRE MULTIVALUED FUNCTION, RETRIEVE FIRST VALUE          LVDLET31
      IVAL=NODSPC(ISADD)                                                LVDLET32
  5   NXTADD=LSTSPC(ISADD)                                              LVDLET33
      NODSPC(ISADD)=NODSPC(REGASP)                                      LVDLET34
      LSTSPC(ISADD)=REGASP                                              LVDLET35
      LNKSPC(ISADD)=0                                                   LVDLET36
      FLGSPC(ISADD)=0                                                   LVDLET37
      LSTSPC(NODSPC(REGASP))=ISADD                                      LVDLET38
      NODSPC(REGASP)=ISADD                                              LVDLET39
      IF((FLGSPC(NXTADD).AND.FL0MSK).NE.0) GO TO 2                      LVDLET40
      ISADD=NXTADD                                                      LVDLET41
      GO TO 5                                                           LVDLET42
C     FUNCTION IS SINGLE VALUED, RETRIEVE VALUE                         LVDLET43
   6  IVAL=LSTSPC(IADD)                                                 LVDLET44
   2  IF((FLGSPC(IADD).AND.FL5MSK).EQ.0) GO TO 68                       LVDLET45
      NXFUNC=LNKSPC(IADD)                                               LVDLET46
      IF((FLGSPC(NXFUNC).AND.FL5MSK).NE.0)GO TO 10                      LVDLET47
      NODSPC(IADD)=NODSPC(NXFUNC)                                       LVDLET48
      LSTSPC(IADD)=LSTSPC(NXFUNC)                                       LVDLET49
      LNKSPC(IADD)=LNKSPC(NXFUNC)                                       LVDLET50
      FLGSPC(IADD)=FLGSPC(NXFUNC)                                       LVDLET51
      FLGSPC(IADD)=FLGSPC(IADD).OR.FL5MSK                               LVDLET52
      IF((FLGSPC(IADD).AND.FL0MSK).EQ.0) GO TO 9                        LVDLET53
      KVAL=LSTSPC(IADD)                                                 LVDLET54
  8   KVAL=LSTSPC(KVAL)                                                 LVDLET55
      IF((FLGSPC(LSTSPC(KVAL)).AND.FL0MSK).EQ.0)  GO TO 8               LVDLET56
      LSTSPC(KVAL)=IADD                                                 LVDLET57
  9   IADD=NXFUNC                                                       LVDLET58
```

```
 10   NODSPC(IADD)=NODSPC(REGASP)                                          LVDLET59
      LSTSPC(IADD)=REGASP                                                  LVDLET60
      LNKSPC(IADD)=0                                                       LVDLET61
      FLGSPC(IADD)=0                                                       LVDLET62
      NODSPC(LSTSPC(IADD))=IADD                                            LVDLET63
      LSTSPC(NODSPC(IADD))=IADD                                            LVDLET64
      RETURN                                                               LVDLET65
 72   IF((FLGSPC(IADD).AND.FLOMSK).NE.0)  GO TO 20                         LVDLET66
      IF(IPOS.NE.1) GO TO 99                                               LVDLET67
      IF(ITYP.EQ.3) GO TO 6                                                LVDLET68
      ISTYP=(FLGSPC(IADD).AND.FLG67)                                       LVDLET69
      IF(ISTYP.EQ.ITYP) GO TO 6                                            LVDLET70
 99   IVAL=-1                                                              LVDLET71
      RETURN                                                               LVDLET72
 20   IND=0                                                                LVDLET73
      FLGSPC(IADD)=FLGSPC(IADD).OR.FL4MSK                                  LVDLET74
      LAST=IADD                                                            LVDLET75
      IF(JPOS)121,99,21                                                    LVDLET76
121   LAST1=LNKSPC(LSTSPC(IADD))                                           LVDLET77
      THIS=LAST1                                                           LVDLET78
      GO TO 27                                                             LVDLET79
 21   IF(JPOS.LT.0) GO TO 80                                               LVDLET80
      THIS=LSTSPC(LAST)                                                    LVDLET81
      IF((FLGSPC(THIS).AND.FLOMSK).NE.0) GO TO 99                          LVDLET82
      GO TO 27                                                             LVDLET83
 80   THIS=LNKSPC(LAST)                                                    LVDLET84
      IF(THIS.EQ.LAST1) GO TO 99                                           LVDLET85
 27   IF(ITYP.EQ.3) GO TO 23                                               LVDLET86
      ISTYP=(FLGSPC(THIS).AND.FLG67)                                       LVDLET87
      IF(ISTYP.EQ.ITYP)    GO TO 23                                        LVDLET88
 22   LAST=THIS                                                            LVDLET89
      GO TO 21                                                             LVDLET90
 23   IND=IND+1                                                            LVDLET91
      IF(IND.NE.IPOS)      GO TO 22                                        LVDLET92
C     RETRIEVE THE IPOS(TH OF THE KTYP(TH VALUE BEFORE DELETING            LVDLET93
      IVAL=NODSPC(THIS)                                                    LVDLET94
      MADD=IADD                                                            LVDLET95
      IF(JPOS.GT.0) GO TO 55                                               LVDLET96
      NEXT=LNKSPC(THIS)                                                    LVDLET97
      IF(THIS.EQ.LAST1) GO TO 82                                           LVDLET98
      LNKSPC(LAST)=NEXT                                                    LVDLET99
      GO TO 83                                                             LVDLE100
 82   LNKSPC(LSTSPC(IADD))=NEXT                                            LVDLE101
 83   IF(NEXT.EQ.LAST1) GO TO 84                                           LVDLE102
      LSTSPC(NEXT)=LAST                                                    LVDLE103
      GO TO 85                                                             LVDLE104
 84   LSTSPC(IADD)=LAST                                                    LVDLE105
 85   IADD=THIS                                                            LVDLE106
      GO TO 86                                                             LVDLE107
 55   NEXT=LSTSPC(THIS)                                                    LVDLE108
      IF((FLGSPC(NEXT).AND.FLOMSK).NE.0) GO TO 50                          LVDLE109
      LNKSPC(NEXT)=LAST                                                    LVDLE110
      GO TO 24                                                             LVDLE111
 50   LNKSPC(LSTSPC(IADD))=LAST                                            LVDLE112
 24   IADD=THIS                                                            LVDLE113
      JLAST=LNKSPC(THIS)                                                   LVDLE114
      IF((FLGSPC(LSTSPC(JLAST)).AND.FLOMSK).NE.0) LNKSPC(NEXT)=JLAST       LVDLE115
```

```
      LSTSPC(LAST)=NEXT                                                 LVDLE116
   86 KLAST=LSTSPC(MADD)                                                LVDLE117
      IF(LNKSPC(KLAST).NE.KLAST) GO TO 10                               LVDLE118
C                                                                       LVDLE119
C     CONVERT TO SINGLE VALUE LIST                                      LVDLE120
C                                                                       LVDLE121
      LSTSPC(MADD)=NODSPC(KLAST)                                        LVDLE122
      FLGSPC(MADD)=(FLGSPC(MADD).OR.FLGSPC(KLAST)).AND.NFLG02           LVDLE123
      FLGSPC(KLAST)=0                                                   LVDLE124
      LNKSPC(KLAST)=0                                                   LVDLE125
      NODSPC(KLAST)=NODSPC(REGASP)                                      LVDLE126
      LSTSPC(KLAST)=REGASP                                              LVDLE127
      NODSPC(LSTSPC(KLAST))=KLAST                                       LVDLE128
      LSTSPC(NODSPC(KLAST))=KLAST                                       LVDLE129
      GO TO 10                                                          LVDLE130
   74 IF(((FLGSPC(IADD).AND.FL0MSK).NE.0).OR.((FLGSPC(IADD).AND.FL2MSK) LVDLE131
     *       .EQ.0))   GO TO 99                                         LVDLE132
      LAST=LNKSPC(IADD)                                                 LVDLE133
      NEXT=LSTSPC(IADD)                                                 LVDLE134
      LSTSPC(LAST)=NEXT                                                 LVDLE135
      IF((FLGSPC(NEXT).AND.FL0MSK).NE.0)   GO TO 10                     LVDLE136
      LNKSPC(NEXT)=LAST                                                 LVDLE137
      GO TO 10                                                          LVDLE138
   68 NEXT=LNKSPC(IADD)                                                 LVDLE139
      NEXT1=NEXT                                                        LVDLE140
   25 IF(LNKSPC(NEXT1).EQ.IADD)  GO TO 26                               LVDLE141
      NEXT1=LNKSPC(NEXT1)                                               LVDLE142
      GO TO 25                                                          LVDLE143
   26 LNKSPC(NEXT1)=NEXT                                                LVDLE144
      KONFLC=1                                                          LVDLE145
      GO TO 10                                                          LVDLE146
      END                                                               LVDLE147
```

FORTRAN Version

```
      SUBROUTINE LVEXIT(N)                                                  LVEXIT 2
      COMMON/LVARGS/LVFUNC,LVVARG,LVVAD,LVVPOS,LVVTYP,       LVVAL,
     +LVHEAD,LVVNVL,LVDEST,LVVALS(10),LVTYPE(10),LVSKIP
      COMMON/LVTABL/LVTSIZ,LVMAP(    1)/LVVSEQ/LVSIZE,LVSOSP(     1)
      COMMON /TYP/ NN(3),ERRFLG                                             LVEXIT 3
      COMMON /STRING/ NTYPE,NSTR                                            LVEXIT 4
      COMMON /NEED/ START,ASSOC,LEVEL,STOP                                  LVEXIT 5
      INTEGER R,RTEMP,STJ,STACK,ASSOC,START,STOP                            LVEXIT 6
      COMMON /GIRL/ MM(19),OPRAND                                           LVEXIT 7
      COMMON /HL/ HOL,ACTION,FUNC1,FUNC2,FUNC3,LEFT,RIGHT,STRING,MAXJ       LVEXIT 8
      COMMON /NTIMES/ NTIMES,MAXI                                           LVEXIT 9
      COMMON/NEEDS/STJ,JSTACK,R,JAS,J,JLAST,RTEMP,STACK(400)                LVEXIT10
      INTEGER STRING,HOL,ACTION,RIGHT,FUNC1,FUNC2,FUNC3,OPRAND              LVEXIT11
      LOGICAL ERRFLG                                                        LVEXIT12
      GO TO 25000
25001 CONTINUE
      IF(MAXJ .NE. 0) PRINT 100,MAXJ                                        LVEXIT14
  100 FORMAT(1X,44H STATEMENT IS TOO COMPLEX. CORRECT TO CHAR. ,I3)         LVEXIT15
      IF(MAXJ .EQ. 0) PRINT 200,MAXI,NSTR                                   LVEXIT16
  200 FORMAT(1X,29H STATEMENT TOO LONG AT CHAR. ,I3,3H OF,I3)               LVEXIT17
C COMMENTS USED IN CASE OF GIRS PROBLEMS WHEN MEMORY USED UP                LVEXIT18
C     GO TO 10                                                              LVEXIT19
C     IF(MAXJ .EQ. 0) GO TO 50                                              LVEXIT20
      IF(MAXJ .EQ. 0) GO TO 10                                              LVEXIT21
      DO 30 NCHAR=1,MAXJ                                                    LVEXIT22
      LVVPOS =       NCHAR
      LVVTYP =   3
      LVFUNC=         HOL
      LVVARG=     STRING
      CALL LVFIND(LV2       A,LV2       B,LV2       C,LV2       D)
      LV1    AAD =      STRING
      IF (LVVAL.NE.-1) LV1    AAD = LVVAL
      LV1    AAI = LV1    AAD
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=       LEFT
      LVVARG=LV1    AAI
      CALL LVDLET
      LV1    AAI = LV1    AAD
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=      RIGHT
      LVVARG=LV1    AAI
      CALL LVDLET
      LV1    AAI = LV1    AAD
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=        HOL
      LVVARG=LV1    AAI
      CALL LVDLET
      LV1    AAI = LV1    AAD
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
```

```
      LVFUNC=     STRING
      LVVARG=LV1     AAI
      CALL LVDLET
   30 CONTINUE                                                          LVEXIT24
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=     STRING
      LVVARG=     STRING
      CALL LVDLET
      LV1     AAD =      OPRAND
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=     OPRAND
      LVVARG=LV1     AAD
      CALL LVDLET
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=     STRING
      LVVARG=LV1     AAD
      CALL LVDLET
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=     ACTION
      LVVARG=LV1     AAD
      CALL LVDLET
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=      FUNC1
      LVVARG=LV1     AAD
      CALL LVDLET
   10 CONTINUE                                                          LVEXIT27
      REWIND 19                                                         LVEXIT28
      NTIMES=0                                                          LVEXIT29
C  10 CONTINUE                                                          LVEXIT30
C     REWIND 99                                                         LVEXIT31
C     NTIMES=0                                                          LVEXIT32
      J=NSTR+1                                                          LVEXIT33
      P=STOP                                                            LVEXIT34
      STJ=R                                                             LVEXIT35
      ERRFLG=.TRUE.                                                     LVEXIT36
      JSTACK=1                                                          LVEXIT37
      STACK(JSTACK)=SHIFT(STOP,45) .OR. SHIFT(100,30) .OR. SHIFT(J,15)  LVEXIT38
C     NSTR=0                                                            LVEXIT39
      NSTR=MAXJ                                                         LVEXIT40
      RETURN
25000 CONTINUE
      LV2      A=LV2      B=LV2      C=LV2      D=0
      GO TO 25001
      END
```

GIRL Version

```
$       SUBROUTINE LVEXIT(N)                                                    LVEXIT      2
        COMMON /TYP/ NN(3),ERRFLG                                               LVEXIT      3
        COMMON /STRING/ NTYPE,NSTR                                              LVEXIT      4
        COMMON /NEED/ START,ASSOC,LEVEL,STOP                                    LVEXIT      5
        INTEGER R,RTEMP,STJ,STACK,ASSOC,START,STOP                              LVEXIT      6
        COMMON /GIRL/ MM(19),OPRAND                                             LVEXIT      7
        COMMON /HL/ HOL,ACTION,FUNC1,FUNC2,FUNC3,LEFT,RIGHT,STRING,MAXJ         LVEXIT      8
        COMMON /NTIMES/ NTIMES,MAXI                                             LVEXIT      9
        COMMON/NEEDS/STJ,JSTACK,R,JAS,J,JLAST,RTEMP,STACK(400)                  LVEXIT     10
        INTEGER STRING,HOL,ACTION,RIGHT,FUNC1,FUNC2,FUNC3,OPRAND                LVEXIT     11
        LOGICAL ERRFLG                                                          LVEXIT     12
G       EXECUTE                                                                 LVEXIT     13
        IF(MAXJ .NE. 0) PRINT 100,MAXJ                                          LVEXIT     14
  100 FORMAT(1X,44H STATEMENT IS TOO COMPLEX. CORRECT TO CHAR. ,I3)             LVEXIT     15
        IF(MAXJ .EQ. 0) PRINT 200,MAXI,NSTR                                     LVEXIT     16
  200 FORMAT(1X,29H STATEMENT TOO LONG AT CHAR. ,I3,3H OF,I3)                   LVEXIT     17
C COMMENTS USED IN CASE OF GIRS PROBLEMS WHEN MEMORY USED UP                    LVEXIT     18
C       GO TO 10                                                                LVEXIT     19
C       IF(MAXJ .EQ. 0) GO TO 50                                                LVEXIT     20
        IF(MAXJ .EQ. 0) GO TO 10                                                LVEXIT     21
        DO 30 NCHAR=1,MAXJ                                                      LVEXIT     22
G       STRING+HOL.NCHAR(-LEFT,-RIGHT,-HOL,-STRING)                             LVEXIT     23
   30 CONTINUE                                                                  LVEXIT     24
G       STRING-STRING                                                           LVEXIT     25
G       OPRAND(-OPRAND,-STRING,-ACTION,-FUNC1)                                  LVEXIT     26
   10 CONTINUE                                                                  LVEXIT     27
        REWIND 19                                                               LVEXIT     28
        NTIMES=0                                                                LVEXIT     29
C  10 CONTINUE                                                                  LVEXIT     30
C       REWIND 99                                                               LVEXIT     31
C       NTIMES=0                                                                LVEXIT     32
        J=NSTR+1                                                                LVEXIT     33
        R=STOP                                                                  LVEXIT     34
        STJ=R                                                                   LVEXIT     35
        ERRFLG=.TRUE.                                                           LVEXIT     36
        JSTACK=1                                                                LVEXIT     37
        STACK(JSTACK)=SHIFT(STOP,45) .OR. SHIFT(100,30) .OR. SHIFT(J,15)        LVEXIT     38
C       NSTR=0                                                                  LVEXIT     39
        NSTR=MAXJ                                                               LVEXIT     40
G       COMPLETE                                                                LVEXIT     41
```

```
      SUBROUTINE LVFECH(N)                                              LVFECH 2
      INTEGER FLGSPC,SEQSPC,REGASP                                      LVFECH 3
      COMMON /LVTABL/ MAPSZE,MAP(1) /LVVSEQ/ ISEQSZ,SEQSPC(1)           LVFECH 4
      COMMON/LVVTR1/MEMSZE,REGASP,NODSPC(    1)/LVVTR2/LSTSPC(    1)/   LVFECH 5
     *LVVTR3/LNKSPC(    1)/LVVTR4/FLGSPC(    1)                        LVFECH 6
      COMMON/LVRAND/ KPRIME,KS,KX,KDY,KDX,KTEMP                         LVFECH 7
      READ(N) MEMSZE,REGASP,KPRIME,KS,KX,KTEST,KDY,KTEMP,KDX,KNUM       LVFECH 8
     +,ISEQSZ                                                           LVFECH 9
      READ(N)(NODSPC(I),I=1,MEMSZE)                                     LVFECH10
      READ(N)(LSTSPC(I),I=1,MEMSZE)                                     LVFECH11
      READ(N)(LNKSPC(I),I=1,MEMSZE)                                     LVFECH12
      READ(N)(FLGSPC(I),I=1,MEMSZE)                                     LVFECH13
      READ(N)(SEQSPC(I),I=1,ISEQSZ)                                     LVFECH14
      PRINT 10                                                          LVFECH15
   10 FORMAT(1H ,*  GRAPH HAS BEEN PLACED INTO MEMORY*,//)              LVFECH16
      RETURN                                                            LVFECH17
      END                                                               LVFECH18
```

```
      SUBROUTINE LVFIND(INDEX,INDXAD,KFUNC,KARG)                        LVFIND 2
      COMMON/LVARGS/IFUNC,IARG,IADD,IPOS,ITYP,IVAL,LSTHED,NVAL,         LVFIND 3
     +  IDSTRY,IVALS(10),ITYP1(10),NSKIP.                              LVFIND 4
      INTEGER FLGSPC,REGASP,FL0MSK,FL1MSK,FL2MSK,FL3MSK,FL4MSK,FL5MSK,  LVFIND 5
     + FLG67,SEQSPC                                                     LVFIND 6
      COMMON/LVFLAG/FL0MSK,FL1MSK,FL2MSK,FL3MSK,FL4MSK,FL5MSK,FLG67     LVFIND 7
      COMMON /LVTABL/ MAPSZE,MAP(1) /LVVSEQ/ ISEQS7,SEQSPC(1)           LVFIND 8
      COMMON/LVVTR1/MEMSZE,REGASP,NODSPC(   1)/LVVTR2/LSTSPC(   1)/     LVFIND 9
     *LVVTR3/LNKSPC(   1)/LVVTR4/FLGSPC(   1)                           LVFIND10
      DATA NFLAG4/3678/                                                 LVFIND11
      IADD=IFUNC+IARG                                                   LVFIND12
      IF(IADD.GT.MEMSZE) IADD=IADD-MEMSZE                               LVFIND13
      LSTHED=0                                                          LVFIND14
      IF((FLGSPC(IADD).AND.FL5MSK).EQ.0) GO TO 99                       LVFIND15
    1 IF(NODSPC(IADD).EQ.IARG)  GO TO 4                                 LVFIND16
      IADD=LNKSPC(IADD)                                                 LVFIND17
      IF((FLGSPC(IADD).AND.FL5MSK).NE.0) GO TO 99                       LVFIND18
      GO TO 1                                                           LVFIND19
    4 IF((FLGSPC(IADD).AND.FL0MSK).NE.0)  GO TO 14                      LVFIND20
      ISTYP=(FLGSPC(IADD).AND.FLG67)                                    LVFIND21
      IF(ITYP.EQ.3) GO TO 11                                            LVFIND22
      IF(ISTYP.EQ.3)ISTYP=2                                             LVFIND23
      IF(ISTYP.NE.ITYP)  GO TO 99                                       LVFIND24
   11 IVAL=LSTSPC(IADD)                                                 LVFIND25
      IF((IPOS.NE.1).AND.(IPOS.NE.-1))  GO TO 99                        LVFIND26
      ITYP=(FLGSPC(IADD).AND.FLG67)                                     LVFIND27
      LSTHED=-1                                                         LVFIND28
      RETURN                                                            LVFIND29
   14 LSTHED=IADD                                                       LVFIND30
      IND=0                                                             LVFIND31
      KNDEX=IABS(INDEX)                                                 LVFIND32
      JPOS=IABS(IPOS)                                                   LVFIND33
      IF(NSKIP.EQ.1) GO TO 50                                           LVFIND34
      IF((KFUNC.NE.IFUNC).OR.(KARG.NE.IARG)) GO TO 50                   LVFIND35
      IF((FLGSPC(LSTHED).AND.FL4MSK).NE.0) GO TO 50                     LVFIND36
      IF((IPOS*INDEX).LE.0) GO TO 50                                    LVFIND37
      IF(JPOS.LT.2) GO TO 50                                            LVFIND38
      NDX=FLGSPC(INDXAD)                                                LVFIND39
      IF((NDX.AND.FL5MSK).NE.0)  GO TO 50                               LVFIND40
      IF((NDX.AND.FL1MSK).EQ.0)  GO TO 50                               LVFIND41
      IF(JPOS.GE.KNDEX)  GO TO 25                                       LVFIND42
      IF((JPOS+JPOS).LE.KNDEX)  GO TO 50                                LVFIND43
      IF(IPOS) 30,99,40                                                 LVFIND44
   50 FLGSPC(LSTHED)=FLGSPC(LSTHED).AND.NFLAG4                          LVFIND45
      IF(IPOS) 20,99,10                                                 LVFIND46
C                                                                       LVFIND47
COUNT DOWN FROM THE TOP  OF THE LIST                                    LVFIND48
C                                                                       LVFIND49
   10 IADD=LSTSPC(IADD)                                                 LVFIND50
      IF((FLGSPC(IADD).AND.FL0MSK).NE.0)  GO TO 99                      LVFIND51
      ISTYP=(FLGSPC(IADD).AND.FLG67)                                    LVFIND52
      IF(ITYP.EQ.3) GO TO 22                                            LVFIND53
      IF(ISTYP.EQ.3)ISTYP=2                                             LVFIND54
      IF(ISTYP.NE.ITYP) GO TO 10                                        LVFIND55
   22 IND=IND+1                                                         LVFIND56
      IF(IND.NE.JPOS) GO TO 10                                          LVFIND57
   28 IVAL=NODSPC(IADD)                                                 LVFIND58
```

```
      ITYP=(FLGSPC(IADD).AND.FLG67)                                    LVFIND59
   55 INDEX=IPOS                                                       LVFIND60
      INDXAD=IADD                                                      LVFIND61
      KFUNC=IFUNC                                                      LVFIND62
      KARG=IARG                                                        LVFIND63
      RETURN                                                           LVFIND64
C                                                                      LVFIND65
COUNT UP FROM THE BOTTOM OF THE LIST                                   LVFIND66
C                                                                      LVFIND67
   20 IADD=LSTSPC(IADD)                                                LVFIND68
      KTEST=0                                                          LVFIND69
   23 IADD=LNKSPC(IADD)                                                LVFIND70
      IF(KTEST.EQ.0) GO TO 24                                          LVFIND71
      IF((FLGSPC(LSTSPC(IADD)).AND.FLGMSK).NE.0)  GO TO 99             LVFIND72
   24 KTEST=1                                                          LVFIND73
      ISTYP=(FLGSPC(IADD).AND.FLG67)                                   LVFIND74
      IF(ITYP.EQ.3) GO TO 21                                           LVFIND75
      IF(ISTYP.EQ.3)ISTYP=2                                            LVFIND76
      IF(ISTYP.NE.ITYP) GO TO 23                                       LVFIND77
   21 IND=IND+1                                                        LVFIND78
      IF(IND.NE.JPOS) GO TO 23                                         LVFIND79
   29 IVAL=NODSPC(IADD)                                                LVFIND80
      ITYP=(FLGSPC(IADD).AND.FLG67)                                    LVFIND81
      GO TO 55                                                         LVFIND82
   25 IF(IPOS) 40,99,30                                                LVFIND83
C                                                                      LVFIND84
COUNT DOWN FROM INDXADD                                                LVFIND85
C                                                                      LVFIND86
   30 JPOS=IABS(JPOS-KNDEX)                                            LVFIND87
      IADD=INDXAD                                                      LVFIND88
      IF(JPOS.EQ.0) GO TO 28                                           LVFIND89
      GO TO 10                                                         LVFIND90
C                                                                      LVFIND91
COUNT UP FROM INDXADD                                                  LVFIND92
C                                                                      LVFIND93
   40 JPOS=IABS(JPOS-KNDEX)                                            LVFIND94
      IADD=INDXAD                                                      LVFIND95
      IF(JPOS.EQ.0) GO TO 29                                           LVFIND96
      KTEST=1                                                          LVFIND97
      GO TO 23                                                         LVFIND98
   99 IVAL=-1                                                          LVFIND99
      INDEX=INDXAD=KFUNC=KARG=0                                        LVFIN100
      RETURN                                                           LVFIN101
      END                                                              LVFIN102
```

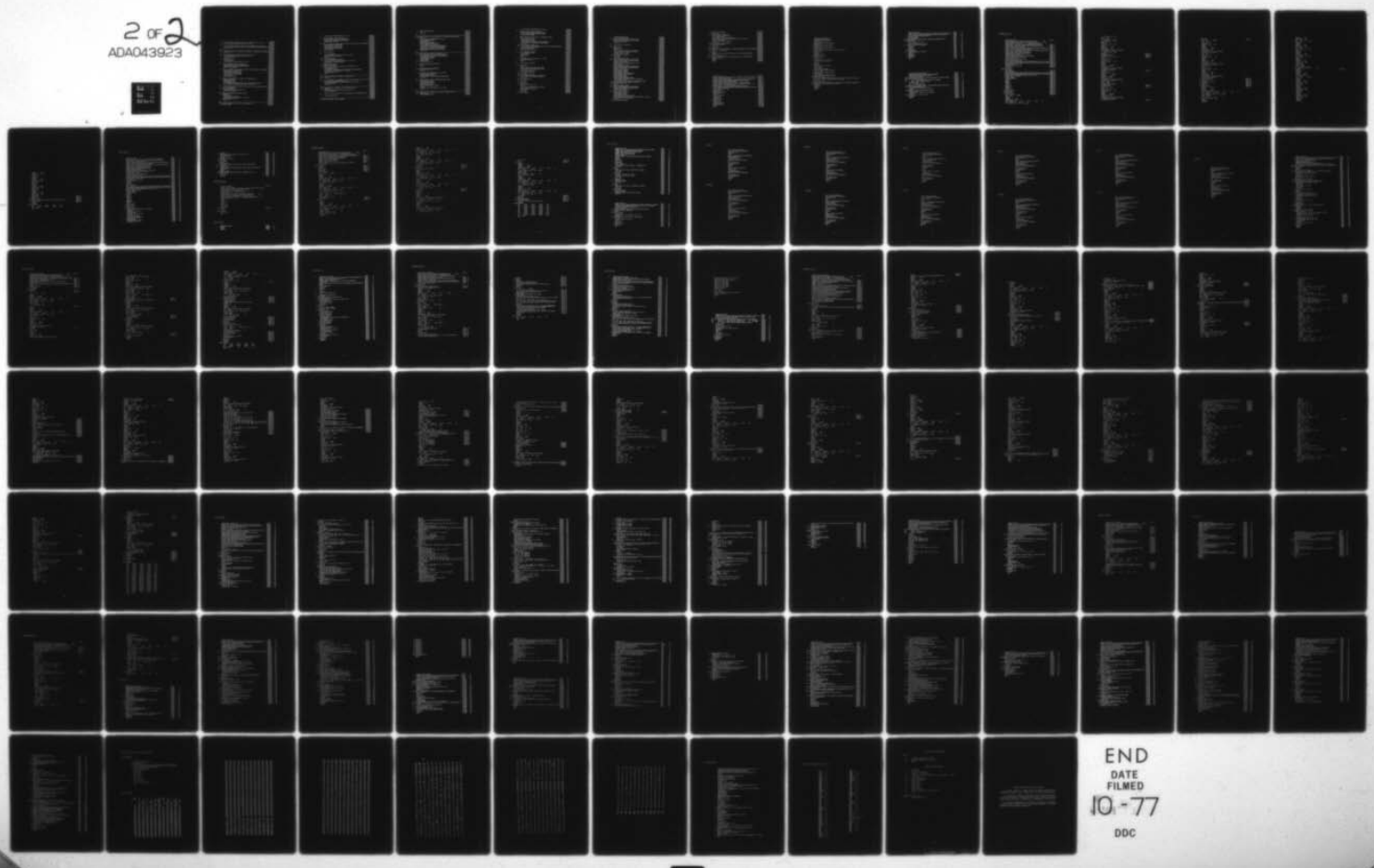
```
      SUBROUTINE LVGRN(NODE)                                            LVGRL  2
      INTEGER FLGSPC,REGASP                                             LVGRL  3
      COMMON/LVVTR1/MEMSZE,REGASP,NODSPC(    1)/LVVTR2/LSTSPC(    1)/  LVGRL  4
     *LVVTR3/LNKSPC(    1)/LVVTR4/FLGSPC(    1)                        LVGRL  5
      COMMON/LVRAND/ KPRIME,KSEED,NROW,KDNODE,KDROW,KTEMP               LVGRL  6
      NODE=KTEMP+KDNODE                                                 LVGRL  7
      KTEMP=NODE                                                        LVGRL  8
      KDNODE=KDNODE+1                                                   LVGRL  9
      IF(NODE.GT.MEMSZE) GO TO 5                                        LVGRL 10
      RETURN                                                            LVGRL 11
C     RESIDUE GENERATION ?                                              LVGRL 12
    5 IF(NROW.GT.KPRIME) GO TO 10                                       LVGRL 13
      NROW=NROW+KSEED                                                   LVGRL 14
      IF(NROW.GT.KPRIME) NROW=NROW-KPRIME                               LVGRL 15
      NODE=NROW                                                         LVGRL 16
      KTEMP=NODE                                                        LVGRL 17
      KDNODE=KPRIME+1                                                   LVGRL 18
C     RESIDUE GENERATION ?                                              LVGRL 19
      IF(NODE.NE.KSEED) RETURN                                          LVGRL 20
      NROW=0                                                            LVGRL 21
      KDROW=KPRIME                                                      LVGRL 22
C     RESIDUE GENERATION                                                LVGRL 23
   10 KDROW=KDROW+1                                                     LVGRL 24
      NROW=NROW+KDROW                                                   LVGRL 25
      NODE=NROW                                                         LVGRL 26
      KTEMP=NODE                                                        LVGRL 27
      KDNODE=KDROW                                                      LVGRL 28
      IF(NODE.GT.MEMSZE) GO TO 20                                       LVGRL 29
      RETURN                                                            LVGRL 30
   20 PRINT 15                                                          LVGRL 31
   15 FORMAT(1H ,*ERROR...NUMBER OF NODES EXCEEDS REQUESTED MEMORY,*/*  LVGRL 32
     *     PROGRAM IS TERMINATED.*)                                     LVGRL 33
      STOP                                                              LVGRL 34
      END                                                               LVGRL 35
```

```
      SUBROUTINE LVNSRT                                                 LVNSRT 2
      COMMON/LVARGS/IFUNC,IARG,IADD,IPOS,ITYP2,IVAL,LSTHED,NVAL,        LVNSRT 3
     +   IDSTRY,IVALS(10),ITYP(10),NSKIP                               LVNSRT 4
      INTEGER FLGSPC,FL0MSK,FL1MSK,FL2MSK,FL5MSK,FLG67,REGASP,TEMP,THIS,LVNSRT 5
     + FLGTMP,TWO,THREE,HEAD,OLDLOC,ASPREG,SEQSPC,FL3MSK,FL4MSK         LVNSRT 6
      COMMON/LVVTR1/MEMSZE,REGASP,NODSPC(    1)/LVVTR2/LSTSPC(    1)/  LVNSRT 7
     *LVVTR3/LNKSPC(    1)/LVVTR4/FLGSPC(    1)                        LVNSRT 8
      COMMON/LVFLAG/FL0MSK,FL1MSK,FL2MSK,FL3MSK,FL4MSK,FL5MSK,FLG67     LVNSRT 9
      COMMON /LVTABL/ MAPSZE,MAP(1) /LVVSEQ/ ISEQSZ,SEQSPC(1)           LVNSRT10
      DATA TWO/29/,THREE/38/,NFLG67/3748/                               LVNSRT11
C                                                                       LVNSRT12
      FLGTMP=FL1MSK                                                     LVNSRT13
C                                                                       LVNSRT14
      IF(REGASP.EQ.LSTSPC(REGASP)) GO TO 98                             LVNSRT15
C                                                                       LVNSRT16
C   FORM FIRST WORD OF SINGLE OR MULTIVALUED FUNCTION                   LVNSRT17
      IF(NVAL.EQ.1)GO TO 20                                             LVNSRT18
      LSTTMP=REGASP                                                     LVNSRT19
      FLGTMP=(FLGTMP.OR.FL2MSK)                                         LVNSRT20
      FLGTMP=(FLGTMP.OR.FL0MSK)                                         LVNSRT21
      GO TO 21                                                          LVNSRT22
   20 LSTTMP=IVALS(1)                                                   LVNSRT23
   21 FLGTMP=(FLGTMP.OR.ITYP(1))                                        LVNSRT24
C                                                                       LVNSRT25
C-----------------------------------------------------------------------LVNSRT26
C-DETERMINE ADDRESS FOR FUNCTION                                        LVNSRT27
      IADD=IFUNC+IARG                                                   LVNSRT28
      IF(IADD.GT.MEMSZE) IADD=IADD-MEMSZE                               LVNSRT29
C                                                                       LVNSRT30
C   IF THAT ADDRESS IS ALREADY IN WORKING SPACE, GO TO 25               LVNSRT31
      IF(IDSTRY-1) 125,300,350                                          LVNSRT32
  125 IF((FL1MSK.AND.FLGSPC(IADD)).NE.0) GO TO 25                       LVNSRT33
C                                                                       LVNSRT34
C    UPDATE REGASP(IF NECESSARY)                                        LVNSRT35
      IF(IADD .EQ. REGASP) REGASP=LSTSPC(IADD)                          LVNSRT36
C                                                                       LVNSRT37
C   UPDATE AVAILABLE SPACE                                              LVNSRT38
      LSTSPC(NODSPC(IADD))=LSTSPC(IADD)                                 LVNSRT39
      NODSPC(LSTSPC(IADD))=NODSPC(IADD)                                 LVNSRT40
C                                                                       LVNSRT41
C    INSERT FUNCTION                                                    LVNSRT42
      NODSPC(IADD)=IARG                                                 LVNSRT43
      LSTSPC(IADD)=LSTTMP                                               LVNSRT44
      LNKSPC(IADD)=IADD                                                 LVNSRT45
C    FLAG 4 IS SET BECAUSE THIS INSERTION MIGHT BE A RECREATION OF AN   LVNSRT46
C    OLD LIST                                                           LVNSRT47
      FLGSPC(IADD)=FLGSPC(IADD).OR.FLGTMP.OR.FL4MSK.OR.FL5MSK           LVNSRT48
C                                                                       LVNSRT49
C   INSERT ANY ADDITIONAL FUNCTION VALUES                               LVNSRT50
      HEAD=IADD                                                         LVNSRT51
      OLDLOC=IADD                                                       LVNSRT52
      IF(NVAL.GT.1)GO TO 50                                             LVNSRT53
C                                                                       LVNSRT54
C   IF LAST CELL OF AVAILABLE SPACE WAS USED, WRITE MESSAGE             LVNSRT55
      IF(REGASP.EQ.LSTSPC(REGASP)) GO TO 909                            LVNSRT56
      IVAL=IABS(IVALS(1))                                               LVNSRT57
      RETURN                                                            LVNSRT58
```

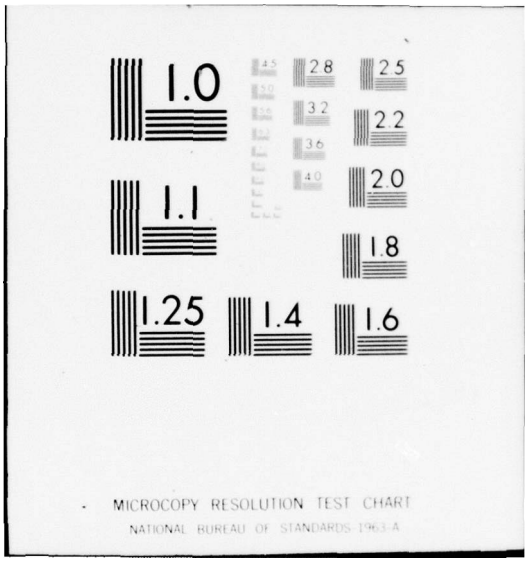
AD-A043 923 DAVID W TAYLOR NAVAL SHIP RESEARCH AND DEVELOPMENT CE--ETC F/G 9/2
MAINTENANCE MANUAL FOR AUDIT. A SYSTEM FOR ANALYZING SESCOMP SO--ETC(U)
AUG 77 R J WYBRANIEC, R REGEN

UNCLASSIFIED DTNSRDC-77-0075-VOL-2 NL

2 OF 2
ADA043923



END
DATE FILMED
10-77
DDC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963-A

```
C                                                                       LVNSRT59
C    IF THAT ADDRESS CONTAINS THE HEAD OF A CONFLICT LIST, GO TO 41     LVNSRT60
  25  IF((FL5MSK.AND.FLGSPC(IADD)).GT.0) GO TO 41                       LVNSRT61
C                                                                       LVNSRT62
C    IF THAT ADDRESS CONTAINS A VALUE ON A MULTIVALUE LIST, GO TO 35    LVNSRT63
      IF((FL2MSK.AND.FLGSPC(IADD)).GT.0.AND.(FL0MSK.AND.FLGSPC(IADD)).EQLVNSRT64
     *.0) GO TO 35                                                      LVNSRT65
C                                                                       LVNSRT66
C-----------------------------------------------------------------------LVNSRT67
C-THE ADDRESS CONTAINS A FUNCTION ON A CONFLICT LIST,BUT NOT THE HEAD OFLVNSRT68
      THIS=IADD                                                         LVNSRT69
C                                                                       LVNSRT70
C    FIND THE PRECEDING FUNCTION ON THE CONFLICT LIST                   LVNSRT71
 26    IF(LNKSPC(LNKSPC(THIS)).EQ.IADD)GO TO 27                        LVNSRT72
       THIS=LNKSPC(THIS)                                               LVNSRT73
       GO TO 26                                                        LVNSRT74
 27    LAST=LNKSPC(THIS)                                               LVNSRT75
       NEWLOC=REGASP                                                   LVNSRT76
       IF(REGASP.EQ.LSTSPC(REGASP)) GO TO 98                           LVNSRT77
C                                                                       LVNSRT78
C    UPDATE AVAILABLE SPACE AND REGASP                                  LVNSRT79
       LSTSPC(NODSPC(REGASP,)=LSTSPC(REGASP)                           LVNSRT80
       NODSPC(LSTSPC(REGASP))=NODSPC(REGASP)                           LVNSRT81
       REGASP=LSTSPC(REGASP)                                           LVNSRT82
                                                                       LVNSRT83
C    MOVE THE FUNCTION ON A CONFLICT LIST TO THE FIRST CELL OF AVAILABLE LVNSRT84
       NODSPC(NEWLOC)=NODSPC(IADD)                                     LVNSRT85
       LSTSPC(NEWLOC)=LSTSPC(IADD)                                     LVNSRT86
       LNKSPC(NEWLOC)=LNKSPC(IADD)                                     LVNSRT87
       FLGSPC(NEWLOC)=FLGSPC(IADD)                                     LVNSRT88
       FLGSPC(IADD)=0                                                  LVNSRT89
       LNKSPC(LAST)=NEWLOC                                             LVNSRT90
C                                                                      LVNSRT91
C    INSERT THIS FUNCTION AS THE HEAD OF A CONFLICT LIST               LVNSRT92
       NODSPC(IADD)=IARG                                               LVNSRT93
       LNKSPC(IADD)=IADD                                               LVNSRT94
       LSTSPC(IADD)=LSTTMP                                             LVNSRT95
       FLGSPC(IADD)=FLGSPC(IADD).OR.FLGTMP.OR.FL4MSK.OR.FL5MSK         LVNSRT96
       IF((FLGSPC(NEWLOC).AND.FL0MSK).EQ.0)  GO TO 34                  LVNSRT97
C                                                                      LVNSRT98
C    IF THE FUNCTION THAT WAS MOVED IS THE HEAD OF A MULTIVALUE LIST,FIX LVNSRT99
       NEXT=LSTSPC(NEWLOC)                                             LVNSR100
 30    NEXT=LSTSPC(NEXT)                                               LVNSR101
       IF(LSTSPC(NEXT).NE.IADD)GO TO 30                                LVNSR102
       LSTSPC(NEXT)=NEWLOC                                             LVNSR103
C                                                                      LVNSR104
C    INSERT ANY ADDITIONAL FUNCTION VALUES                             LVNSR105
  34   HEAD=IADD                                                       LVNSR106
       OLDLOC=IADD                                                     LVNSR107
       IF(NVAL.GT.1)GO TO 50                                           LVNSR108
       IF(REGASP.EQ.LSTSPC(REGASP)) GO TO 909                          LVNSR109
       IVAL=IABS(IVALS(1))                                             LVNSR110
       RETURN                                                          LVNSR111
C                                                                      LVNSR112
C-----------------------------------------------------------------------LVNSR113
C-THE ADDRESS CONTAINS A VALUE ON A MULTIVALUE LIST                    LVNSR114
  35   NEWLOC=REGASP                                                   LVNSR115
```

```
      IF(REGASP.EQ.LSTSPC(REGASP)) GO TO 98                                  LVNSR116
C                                                                            LVNSR117
C   UPDATE AVAILABLE SPACE AND REGASP                                        LVNSR118
      LSTSPC(NODSPC(REGASP))=LSTSPC(REGASP)                                  LVNSR119
      NODSPC(LSTSPC(REGASP))=NODSPC(REGASP)                                  LVNSR120
      REGASP=LSTSPC(REGASP)                                                  LVNSR121
C                                                                            LVNSR122
C   MOVE THE VALUE ON A MULTIVALUE LIST TO THE FIRST CELL OF AVAILABLE SL    LVNSR123
      NODSPC(NEWLOC)=NODSPC(IADD)                                            LVNSR124
      LSTSPC(NEWLOC)=LSTSPC(IADD)                                            LVNSR125
      LNKSPC(NEWLOC)=LNKSPC(IADD)                                            LVNSR126
      FLGSPC(NEWLOC)=FLGSPC(IADD)                                            LVNSR127
      FLGSPC(IADD)=0                                                         LVNSR128
C                                                                            LVNSR129
C     RESET POINTERS                                                         LVNSR130
C                                                                            LVNSR131
      L1=LSTSPC(NEWLOC)                                                      LVNSR132
      IF((FLOMSK.AND.FLGSPC(L1)).EQ.0) GO TO 200                             LVNSR133
      LNKSPC(LSTSPC(L1))=NEWLOC                                              LVNSR134
      GO TO 201                                                              LVNSR135
 200  LNKSPC(L1)=NEWLOC                                                      LVNSR136
 201  KZVAL=LSTSPC(LNKSPC(NEWLOC))                                          LVNSR137
      IF((FLGSPC(KZVAL).AND.FLOMSK).NE.0)  GO TO 38                          LVNSR138
      LSTSPC(LNKSPC(NEWLOC))=NEWLOC                                          LVNSR139
      GO TO 39                                                               LVNSR140
  38  LSTSPC(KZVAL)=NEWLOC                                                   LVNSR141
  39  NODSPC(IADD)=IARG                                                      LVNSR142
C   INSERT THIS FUNCTION AS THE HEAD OF A CONFLICT LIST                      LVNSR143
      LNKSPC(IADD)=IADD                                                      LVNSR144
      LSTSPC(IADD)=LSTTMP                                                    LVNSR145
      FLGSPC(IADD)=FLGSPC(IADD).OR.FLGTMP.OR.FL4MSK.OR.FL5MSK                LVNSR146
      IF(REGASP.EQ.LSTSPC(REGASP)) GO TO 909                                 LVNSR147
      IVAL=IABS(IVALS(1))                                                    LVNSR148
      RETURN                                                                 LVNSR149
C                                                                            LVNSR150
C-------------------------------------------------------------------------LVNSR151
C-THE ADDRESS CONTAINS THE HEAD OF A CONFLICT LIST                           LVNSR152
 41   THIS=IADD                                                              LVNSR153
C                                                                            LVNSR154
C   IF THE FUNCTION TO BE INSERTED IS NOT ON THE CONFLICT LIST, GO TO 60     LVNSR155
  42  IF(NODSPC(THIS).EQ.IARG)GO TO 43                                       LVNSR156
      IF((FLGSPC(LNKSPC(THIS)).AND.FL5MSK).NE.0) GO TO 60                    LVNSR157
      THIS=LNKSPC(THIS)                                                      LVNSR158
      GO TO 42                                                               LVNSR159
C                                                                            LVNSR160
C-------------------------------------------------------------------------LVNSR161
C-THE FUNCTION TO BE INSERTED IS ON THE CONFLICT LIST                        LVNSR162
  43  HEAD=THIS                                                              LVNSR163
      IF((FLOMSK.AND.FLGSPC(THIS)).EQ.0) GO TO 51                            LVNSR164
      NEXT=LSTSPC(THIS)                                                      LVNSR165
C                                                                            LVNSR166
C     OLDLOC IS THE LOCATION OF THE LAST VALUE ON THE MULTIVALUE LIST        LVNSR167
C                                                                            LVNSR168
      OLDLOC=LNKSPC(NEXT)                                                    LVNSR169
C                                                                            LVNSR170
C-------------------------------------------------------------------------LVNSR171
C-INSERT ADDITIONAL FUNCTION VALUES                                          LVNSR172
```

```
 50   LSTASP=NODSPC(REGASP)                                              LVNSR173
      IN=0                                                               LVNSR174
      GO TO 56                                                           LVNSR175
C                                                                        LVNSR176
C-------------------------------------------------------------------------LVNSR177
C-FORM MULTIVALUE LIST TO ADD VALUE(S) TO SINGLE-VALUED FUNCTION         LVNSR178
  51  IN=0                                                               LVNSR179
      IF(REGASP.EQ.LSTSPC(REGASP))GO TO 98                               LVNSR180
      LSTASP=NODSPC(REGASP)                                              LVNSR181
      NEWLOC=REGASP                                                      LVNSR182
      REGASP=LSTSPC(REGASP)                                              LVNSR183
      NODSPC(NEWLOC)=LSTSPC(THIS)                                        LVNSR184
      TEMP=(FLGSPC(THIS).AND.FLG67)                                      LVNSR185
      FLGSPC(NEWLOC)=(TEMP.OR.FLGSPC(NEWLOC))                            LVNSR186
      FLGSPC(THIS)=(FLGSPC(THIS).AND.NFLG67)                             LVNSR187
      FLGSPC(THIS)=(FL2MSK.OR.FLGSPC(THIS))                              LVNSR188
      FLGSPC(THIS)=(FL0MSK.OR.FLGSPC(THIS))                              LVNSR189
      OLDLOC=THIS                                                        LVNSR190
C                                                                        LVNSR191
C-------------------------------------------------------------------------LVNSR192
C   INSERT ANOTHER VALUE ON MULTIVALUE LIST                              LVNSR193
 52   FLGSPC(NEWLOC)=(FL2MSK.OR.FLGSPC(NEWLOC))                          LVNSR194
      FLGSPC(NEWLOC)=(FL1MSK.OR.FLGSPC(NEWLOC))                          LVNSR195
      LSTSPC(OLDLOC)=NEWLOC                                              LVNSR196
      LNKSPC(NEWLOC)=OLDLOC                                              LVNSR197
      OLDLOC=NEWLOC                                                      LVNSR198
 56   NEWLOC=REGASP                                                      LVNSR199
      IF(IN.GT.0)GO TO 57                                                LVNSR200
C                                                                        LVNSR201
C   NO VALUES HAVE BEEN INSERTED YET                                     LVNSR202
      IN=1                                                               LVNSR203
      GO TO 58                                                           LVNSR204
C                                                                        LVNSR205
C   SOME VALUES HAVE BEEN INSERTED                                       LVNSR206
 57   IF(IN.EQ.NVAL)GO TO 67                                             LVNSR207
      IN=IN+1                                                            LVNSR208
C                                                                        LVNSR209
   58 IF(REGASP.EQ.LSTSPC(REGASP)) GO TO 909                             LVNSR210
      REGASP=LSTSPC(REGASP)                                              LVNSR211
      NODSPC(NEWLOC)=IVALS(IN)                                           LVNSR212
      FLGSPC(NEWLOC)=(ITYP(IN).OR.FLGSPC(NEWLOC))                        LVNSR213
      GO TO 52                                                           LVNSR214
C                                                                        LVNSR215
C   END MULTIVALUE LIST AND UPDATE AVAILABLE SPACE                       LVNSR216
 67   LSTSPC(OLDLOC)=HEAD                                                LVNSR217
      NODSPC(REGASP)=LSTASP                                              LVNSR218
      LSTSPC(LSTASP)=REGASP                                              LVNSR219
      IVAL=IABS(IVALS(1))                                                LVNSR220
      LNKSPC(LSTSPC(HEAD))=OLDLOC                                        LVNSR221
      NVAL=IN                                                            LVNSR222
      IF(REGASP.EQ.LSTSPC(REGASP)) GO TO 909                             LVNSR223
      RETURN                                                             LVNSR224
C                                                                        LVNSR225
C-------------------------------------------------------------------------LVNSR226
C-THE FUNCTION TO BE INSERTED IS NOT ON THE CONFLICT LIST                LVNSR227
 60   ASPREG=REGASP                                                      LVNSR228
      LSTASP=NODSPC(REGASP)                                              LVNSR229
```

```
      IF(REGASP.EQ.LSTSPC(REGASP)) GO TO 98                              LVNSR230
C                                                                        LVNSR231
C   UPDATE AVAILABLE SPACE AND REGASP                                    LVNSR232
      LSTSPC(NODSPC(REGASP))=LSTSPC(REGASP)                              LVNSR233
      NODSPC(LSTSPC(REGASP))=NODSPC(REGASP)                              LVNSR234
      REGASP=LSTSPC(REGASP)                                              LVNSR235
C                                                                        LVNSR236
C   INSERT FUNCTION IN FIRST CELL OF AVAILABLE SPACE                     LVNSR237
      NODSPC(ASPREG)=IARG                                                LVNSR238
      IF(NVAL.EQ.1)GO TO 611                                             LVNSR239
      LSTSPC(ASPREG)=REGASP                                              LVNSR240
      FLGSPC(ASPREG)=(FL2MSK.OR.FLGSPC(ASPREG))                          LVNSR241
      FLGSPC(ASPREG)=(FL0MSK.OR.FLGSPC(ASPREG))                          LVNSR242
      GO TO 612                                                          LVNSR243
 611  LSTSPC(ASPREG)=IVALS(1)                                            LVNSR244
 612  FLGSPC(ASPREG)=FLGSPC(ASPREG).OR.ITYP(1).OR.FL1MSK.OR.FL4MSK       LVNSR245
      LNKSPC(ASPREG)=LNKSPC(THIS)                                        LVNSR246
      LNKSPC(THIS)=ASPREG                                                LVNSR247
      IF(NVAL.EQ.1)GO TO 613                                             LVNSR248
C                                                                        LVNSR249
C   INSERT ADDITIONAL VALUES                                             LVNSR250
      LSTASP=NODSPC(REGASP)                                              LVNSR251
      OLDLOC=ASPREG                                                      LVNSR252
      HEAD=ASPREG                                                        LVNSR253
      IN=0                                                               LVNSR254
      GO TO 56                                                           LVNSR255
 613  IF(REGASP.EQ.LSTSPC(REGASP)) GO TO 909                             LVNSR256
      IVAL=IABS(IVALS(1))                                                LVNSR257
      RETURN                                                             LVNSR258
C                                                                        LVNSR259
C     DESTRUCTIVE INSERTION                                              LVNSR260
C                                                                        LVNSR261
 350  IADD1=IADD                                                         LVNSR262
      INDEX=0                                                            LVNSR263
      CALL LVFIND(INDEX,INDEX,INDEX,INDEX)                               LVNSR264
      FLGSPC(IADD)=FLGSPC(IADD).OR.FL4MSK                                LVNSR265
      IF(IVAL.EQ.-1) GO TO 90                                            LVNSR266
      IF(LSTHED) 354,90,356                                              LVNSR267
 354  LSTSPC(IADD)=IVALS(1)                                              LVNSR268
      GO TO 365                                                          LVNSR269
 356  NODSPC(IADD)=IVALS(1)                                              LVNSR270
 365  FLGSPC(IADD)=FLGSPC(IADD).AND.NFLG67                               LVNSR271
      FLGSPC(IADD)=FLGSPC(IADD).OR.ITYP(1)                               LVNSR272
      GO TO 360                                                          LVNSR273
  90  IF(IPOS)91,99,92                                                   LVNSR274
  91  IPOS=IPOS+1                                                        LVNSR275
      GO TO 93                                                           LVNSR276
  92  IPOS=IPOS-1                                                        LVNSR277
  93  IADD=IADD1                                                         LVNSR278
      IF(IPOS.EQ.0) GO TO 125                                            LVNSR279
      INDEX=0                                                            LVNSR280
      CALL LVFIND(INDEX,INDEX,INDEX,INDEX)                               LVNSR281
      IF(IVAL.EQ.-1) GO TO 99                                            LVNSR282
      IF(IPOS.LT.0) GO TO 370                                            LVNSR283
      IADD=IADD1                                                         LVNSR284
      GO TO 125                                                          LVNSR285
 370  NEWLOC=REGASP                                                      LVNSR286
```

```
      IF(LSTHED) 325,99,375                                              LVNSR287
C     UPDATE AVAILABLE SPACE                                             LVNSR288
  375 LSTSPC(NODSPC(REGASP))=LSTSPC(REGASP)                              LVNSR289
      NODSPC(LSTSPC(REGASP))=NODSPC(REGASP)                              LVNSR290
      REGASP=LSTSPC(REGASP)                                              LVNSR291
      GO TO 377                                                          LVNSR292
C                                                                        LVNSR293
C     NONDESTRUCTIVE INSERTION                                           LVNSR294
C                                                                        LVNSR295
  300 IADD1=IADD                                                         LVNSR296
      NEWLOC=REGASP                                                      LVNSR297
      INDEX=0                                                            LVNSR298
      CALL LVFIND(INDEX,INDEX,INDEX,INDEX)                               LVNSR299
      FLGSPC(IADD)=FLGSPC(IADD).OR.FL4MSK                                LVNSR300
      IF(IVAL.EQ.-1) GO TO 90                                            LVNSR301
      IF(LSTHED) 344,90,346                                              LVNSR302
  344 IF(IPOS.GT.0) GO TO 325                                            LVNSR303
      IADD=IADD1                                                         LVNSR304
      GO TO 125                                                          LVNSR305
C     CREATE MULTIVALUE LIST                                             LVNSR306
 325  LSTSPC(NODSPC(REGASP))=LSTSPC(REGASP)                              LVNSR307
      NODSPC(LSTSPC(REGASP))=NODSPC(REGASP)                              LVNSR308
      REGASP=LSTSPC(REGASP)                                              LVNSR309
      IF(REGASP.EQ.LSTSPC(REGASP)) GO TO 909                             LVNSR310
      NWLOC2=REGASP                                                      LVNSR311
C     UPDATE AVAILABLE SPACE                                             LVNSR312
      LSTSPC(NODSPC(REGASP))=LSTSPC(REGASP)                              LVNSR313
      NODSPC(LSTSPC(REGASP))=NODSPC(REGASP)                              LVNSR314
      REGASP=LSTSPC(REGASP)                                              LVNSR315
      NODSPC(NEWLOC)=IVALS(1)                                            LVNSR316
      LSTSPC(NEWLOC)=NWLOC2                                              LVNSR317
      LNKSPC(NEWLOC)=NWLOC2                                              LVNSR318
      FLGSPC(NEWLOC)=FLGTMP.OR.FL2MSK                                    LVNSR319
      NODSPC(NWLOC2)=LSTSPC(IADD)                                        LVNSR320
      LSTSPC(NWLOC2)=IADD                                                LVNSR321
      LNKSPC(NWLOC2)=NEWLOC                                              LVNSR322
      KLGTEP=FLGSPC(IADD).AND.FLG67                                      LVNSR323
      FLGSPC(NWLOC2)=(FL1MSK.OR.FL2MSK).OR.KLGTEP                        LVNSR324
      LSTSPC(IADD)=NEWLOC                                                LVNSR325
      FLGSPC(IADD)=(FLGSPC(IADD).OR.FL0MSK).OR.FL2MSK                    LVNSR326
  320 IF(REGASP.EQ.LSTSPC(REGASP)) GO TO 909                             LVNSR327
  360 IVAL=IABS(IVALS(1))                                                LVNSR328
      RETURN                                                             LVNSR329
C     UPDATE AVAILABLE SPACE                                             LVNSR330
 346  LSTSPC(NODSPC(REGASP))=LSTSPC(REGASP)                              LVNSR331
      NODSPC(LSTSPC(REGASP))=NODSPC(REGASP)                              LVNSR332
      REGASP=LSTSPC(REGASP)                                              LVNSR333
      IF(IPOS.LT.0) GO TO 347                                            LVNSR334
 377  ISTLOC=LNKSPC(IADD)                                                LVNSR335
      NODSPC(NEWLOC)=IVALS(1)                                            LVNSR336
      LSTSPC(NEWLOC)=IADD                                                LVNSR337
      LNKSPC(NEWLOC)=ISTLOC                                              LVNSR338
      FLGSPC(NEWLOC)=FLGTMP.OR.FL2MSK                                    LVNSR339
      IF((FLGSPC(LSTSPC(ISTLOC)).AND.FL0MSK).EQ.0) GO TO 321             LVNSR340
      LSTSPC(LSTSPC(ISTLOC))=NEWLOC                                      LVNSR341
      GO TO 322                                                          LVNSR342
  321 LSTSPC(ISTLOC)=NEWLOC                                              LVNSR343
```

```
  322 LNKSPC(IADD)=NEWLOC                                                LVNSR344
      GO TO 320                                                          LVNSR345
  347 NODSPC(NEWLOC)=IVALS(1)                                            LVNSR346
      LSTSPC(NEWLOC)=LSTSPC(IADD)                                        LVNSR347
      LNKSPC(NEWLOC)=IADD                                                LVNSR348
      FLGSPC(NEWLOC)=FLGTMP.OR.FL2MSK                                    LVNSR349
      IF((FLGSPC(LSTSPC(IADD)).AND.FL0MSK).EQ.0) GO TO 323               LVNSR350
      KZVAL=LSTSPC(IADD)                                                 LVNSR351
      LNKSPC(LSTSPC(KZVAL))=NEWLOC                                       LVNSR352
      GO TO 324                                                          LVNSR353
  323 LNKSPC(LSTSPC(IADD))=NEWLOC                                        LVNSR354
  324 LSTSPC(IADD)=NEWLOC                                                LVNSR355
      GO TO 320                                                          LVNSR356
   98 IVAL=-3                                                            LVNSR357
      PRINT 20001                                                        LVNSR358
20001 FORMAT(  * ERROR...THERE IS NO ADDITIONAL SPACE FOR THE GRAPH, THELVNSR359
     * PROGRAM IS TERMINATED*)                                           LVNSR360
      STOP                                                               LVNSR361
   99 IVAL=-1                                                            LVNSR362
   22 FORMAT(1X,I5,1H(,I5,35H) USED LAST CELL OF AVAILABLE SPACE)        LVNSR363
      RETURN                                                             LVNSR364
  909 PRINT 22,IFUNC,IARG                                                LVNSR365
C     THIS INSERTION HAS FILLED GIRS MEMORY - CALL A USER SUPPLIED       LVNSR366
C     PROGRAM - LVEXIT.                                                  LVNSR367
      IVAL=-1                                                            LVNSR368
      RETURN                                                             LVNSR369
      END                                                                LVNSR370
```

```
      SUBROUTINE LVSETP                                                  LVSETP 2
      INTEGER FLGSPC,FLAGSP,REGASP,BINFIL,FL0MSK,FL1MSK,FL2MSK,FL5MSK,   LVSETP 3
     + FL3MSK,FL4MSK,FLG67,SEQSPC                                        LVSETP 4
      COMMON/LVFLAG/FL0MSK,FL1MSK,FL2MSK,FL3MSK,FL4MSK,FL5MSK,FLG67      LVSETP 5
      COMMON/LVVTR5/BINFIL,KOMPAN,NODESP(1)/LVVTR6/LISTSP(1)             LVSETP 6
     + /LVVTR7/LINKSP(1)/LVVTR8/FLAGSP(1)                                LVSETP 7
      COMMON /LVTABL/ MAPSZE,MAP(1) /LVVSEQ/ ISEQSZ,SEQSPC(1)            LVSETP 8
      COMMON/LVVTR1/MEMSZE,REGASP,NODSPC(    1)/LVVTR2/LSTSPC(    1)/    LVSETP 9
     *LVVTR3/LNKSPC(    1)/LVVTR4/FLGSPC(    1)                         LVSETP10
      COMMON/LVRAND/ KPRIME,KSEED,NROW,KDNODE,KDROW,KTEMP                LVSETP11
      DATA FL0MSK/2008/,FL1MSK/1008/,FL2MSK/408/,FL5MSK/48/,FLG67/38/,   LVSETP12
     + FL3MSK/208/,FL4MSK/108/                                           LVSETP13
      KSEED=KPRIME/2                                                     LVSETP14
      NROW=KSEED                                                         LVSETP15
      KTEMP=KSEED-KPRIME                                                 LVSETP16
      KDNODE=KPRIME                                                      LVSETP17
      REGASP=1                                                           LVSETP18
      DO 10 I=2,MEMSZE                                                   LVSETP19
      LNKSPC(I)=0                                                        LVSETP20
      FLGSPC(I)=0                                                        LVSETP21
      NODSPC(I)=I-1                                                      LVSETP22
   10 LSTSPC(I-1)=I                                                      LVSETP23
      FLGSPC(1)=0                                                        LVSETP24
      LNKSPC(1)=0                                                        LVSETP25
      NODSPC(1)=MEMSZE                                                   LVSETP26
      LSTSPC(MEMSZE)=1                                                   LVSETP27
      RETURN                                                             LVSETP28
      END                                                                LVSETP29
```

```
      SUBROUTINE MODIO(MODE)
      DIMENSION IBUF(80),IEND(3)
      IF(MODE .NE. 0) GO TO 5
      WRITE(13,1)
    1 FORMAT(5X,16H OUTPUT DEVICE X)
      WRITE(14,2)
    2 FORMAT(5X,16H OUTPUT DEVICE Y)
      WRITE(15,3)
    3 FORMAT(5X,16H OUTPUT DEVICE Z)
      WRITE(6,6)
    6 FORMAT(1H1,52X,27H RESULTS OF ROLL CALL CHECK)
    5 DO 10 I=10,12
      ENDFILE I
      REWIND I
      READ(I,7) ICHAR
    7 FORMAT(A1)
   10 IEND(I-9)=EOF(I)
      IEOF=0
      DO 15 I=1,3
      IF(IEND(I) .NE. 0) GO TO 15
      IF(IEOF .EQ. 1) GO TO 40
      IEOF=1
      IOUT=9+I
   15 CONTINUE
      IF(IEOF .EQ. 0) GO TO 50
      REWIND IOUT
      IOUT2=IOUT+3
      WRITE(IOUT2,20) MODE
   20 FORMAT(//20X,12H MODE INDEX=,I3)
      DO 30 I=1,100
      READ(IOUT,25) (IBUF(J),J=1,80)
   25 FORMAT(80A1)
      IF(EOF(IOUT) .NE. 0) GO TO 60
   30 WRITE(IOUT2,25) (IBUF(J),J=1,80)
   40 WRITE(6,45) MODE
   45 FORMAT(//21X,86H **ERROR IN ROLL CALL CHECK - MORE THAN 1 OUTPUT D
     *EVICE WAS WRITTEN ON FOR MODE INDEX ,I3,2H**)
      GO TO 60
   50 WRITE(6,55) MODE
   55 FORMAT(//25X,79H **ERROR IN ROLL CALL CHECK - NO OUTPUT DEVICES WE
     *RE WRITTEN ON FOR MODE INDEX ,I3,2H**)
   60 REWIND 10
      REWIND 11
      REWIND 12
      RETURN
      END
```

```
      FUNCTION NEXT(IA)                                                 NEXT     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH     2
     * JPTR,N,M,JTYP,LSTART,NZ,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,       CY58A   88
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH     4
      INTEGER A,BLANK                                                   NEXT     4
      DATA BLANK/1H /                                                   NEXT     5
C** THIS FUNCTION RETURNS THR NEXT NON-BLANK CHARACTER IN THE            NEXT     6
C** INPUT STREAM                                                        NEXT     7
      IF(IA .GT. N) GO TO 15                                            NEXT     8
      DO 10 I=IA,N                                                      NEXT     9
      IF(A(I) .EQ. BLANK) GO TO 10                                      NEXT    10
C** "A(I)" IS THE NEXT CHARACTER                                        NEXT    11
      NEXT=A(I)                                                         NEXT    12
      JPTR=I+1                                                          NEXT    13
      RETURN                                                            NEXT    14
   10 CONTINUE                                                          NEXT    15
      NEXT=BLANK                                                        CY55     1
C** NO MORE CHARACTERS IN STRING                                        NEXT    17
      JPTR=N+1                                                          NEXT    18
      RETURN                                                            NEXT    19
   15 NEXT=BLANK                                                        CY55     2
      JPTR=IA+1                                                         CY55     3
      RETURN                                                            CY55     4
      END                                                               NEXT    20
```

```
      FUNCTION NXTBLK(ILOC,IEND)                                        NXTBLK   2
      COMMON/BASBLK/IBLOCK(2500),NBLOCK,NB,NBRNCH                       CY58A   51
      COMMON/LABELS/STATRA(2,200),NLABEL                                NXTBLK   4
      INTEGER STATRA,BITGET                                             NXTBLK   5
C** THIS ROUTINE RETURNS THE STARTING LOCATION OF THE BASIC BLOCK       NXTBLK   6
C** WHICH A BRANCH POINTS TO                                            NXTBLK   7
C**       ILOC - BASIC BLOCK TABLE LOCATION OF BRANCH                   NXTBLK   8
C**       IEND - END OF CURRENT BLOCK                                   NXTBLK   9
      I=IBLOCK(ILOC)                                                    NXTBLK  10
      IF(I .EQ. 999) GO TO 10                                           NXTBLK  11
      IF(I .EQ. 998) GO TO 5                                            NXTBLK  12
C** BRANCH IS A STATEMENT LABEL - RETRIEVE BASIC BLOCK START FROM       NXTBLK  13
C** THE STATEMENT NUMBER TABLE                                          NXTBLK  14
      NXTBLK=BITGET(STATRA(2,I),36,18)                                  NXTBLK  15
      RETURN                                                            NXTBLK  16
C** BRANCH IS TO NEXT BASIC BLOCK IN TABLE                              NXTBLK  17
    5 NXTBLK=IEND+1                                                     NXTBLK  18
      IF(NXTBLK .GT. NBLOCK) CALL ERROR(38)                             NXTBLK  19
      RETURN                                                            NXTBLK  20
C** RETURN OR STOP - END OF PATH                                        NXTBLK  21
   10 NXTBLK=0                                                          NXTBLK  22
      RETURN                                                            NXTBLK  23
      END                                                               NXTBLK  24
```

FORTRAN Version

```
      SUBROUTINE PARSE                                                  PARSE  2
      COMMON/LVARGS/LVFUNC,LVVARG,LVVAD,LVVPOS,LVVTYP,       LVVAL,
     +LVHEAD,LVVNVL,LVDEST,LVVALS(10),LVTYPE(10),LVSKIP
      COMMON/LVTABL/LVTSIZ,LVMAP(    1)/LVVSEQ/LVSIZE,LVSQSP(     1)
      COMMON/NEEDS/STJ,JSTACK,R,JAS,J,JLAST,RTEMP,STACK(400)            PARSE  3
      COMMON/FUNC/  NARY(5,12),MARGS,IARGS(50),FNCLOC(5),NFUNC          CY58B  1
      COMMON /STRING/ NTYPE,NSTR,STR                                    PARSE  5
      COMMON /GIRL/NTERMS,PLUS,MINUS,SLASH,LPAR,RPAR,COMMA,STAR,EXP,LT, PARSE  6
     +LE,GT,GE,EQ,NE,OR,AND,NOT,EQUALS,OPRAND                           PARSE  7
      COMMON /HL/ HOL,ACTION,FUNC1,FUNC2,FUNC3,LEFT,RIGHT,STRING,MAXJ   PARSE  8
      COMMON /NEED/ START,ASSOC,LEVEL,STOP                              PARSE  9
      COMMON /TYP/ NARRAY,TYPE1,TYPE2,ERRFLG                            PARSE 10
      COMMON/NOPAR/NOPAR,NDEP,NDEPTH,NFLAG                              PARSE 11
      COMMON /NTIMES/ NTIMES,I                                          PARSE 12
      COMMON/VAR/VFOR(15),NUMCHR,NCHFP,CHAR,NDICT                       CY58B  2
      INTEGER TYPE1,TYPE2,START,TYP(3)                                  PARSE 14
      LOGICAL ERRFLG,FAIL                                               PARSE 15
      INTEGER STR(1),STEMP,ST,DICT(19)                                  PARSE 16
      EQUIVALENCE(DICT(1),PLUS)                                         PARSE 17
      INTEGER PLUS,MINUS,SLASH,LPAR,RPAR,COMMA,STAR,EXP,LT,LE,GT,GE,EQ, PARSE 18
     +NE,OR,AND,NOT,EQUALS,OPRAND,ASSOC,LEVEL,STOP,ACTION,HOL,LEFT,RIGHTPARSE 19
     +,STRING,FUNC1,FUNC2,FUNC3                                         PARSE 20
      DATA NTIMES /0/                                                   PARSE 22
      IF(NTIMES .GT. 0) GO TO 3                                         PARSE 23
      NTIMES=1                                                          PARSE 24
      GO TO 25000
25001 CONTINUE
      CALL PHONEY                                                       PARSE 26
      CALL LVFECH(19)                                                   PARSE 27
      READ(19)PLUS,MINUS,SLASH,LPAR,RPAR,COMMA,STAR,EXP,LT,LE,GT,GE,EQ, PARSE 28
     +NE,OR,AND,NOT,EQUALS,OPRAND,ASSOC,LEVEL,STOP,ACTION,HOL,LEFT,RIGHTPARSE 29
     +,STRING,FUNC1,FUNC2,FUNC3,NTERMS,(TYP(I),I=1,3)                   PARSE 30
    3 IF(NSTR .LE. 0) RETURN                                            PARSE 31
      ERRFLG=.FALSE.                                                    PARSE 32
      START=TYP(NTYPE)                                                  PARSE 33
      NARGS=0                                                           PARSE 34
      NFLAG=0                                                           PARSE 35
      MAXJ=0                                                            PARSE 36
      NOPAR=0                                                           PARSE 37
      TYPE1=-1                                                          PARSE 38
      TYPE2=-1                                                          PARSE 39
      NARRAY=-1                                                         PARSE 40
      NDEPTH=0                                                          PARSE 41
      NDEP=0                                                            PARSE 42
      DO 20 I=1,50                                                      PARSE 43
   20 IARGS(I)=0                                                        PARSE 44
      DO 22 I=1,60                                                      PARSE 45
   22 NARY(I)=0                                                         PARSE 46
      DO 10 I=1,NSTR                                                    PARSE 47
      LV1    AAB =      STRING
      LVVPOS =           I
      LVVTYP =   3
      LVFUNC=         HOL
      LVVARG= LV1    AAB
      CALL LVFIND(LV2       A,LV2       B,LV2       C,LV2       D)
      LV1    AAC = LV1    AAB
      IF (LVVAL.NE.-1) LV1    AAC = LVVAL
```

```
      NTEMP = LV1     AAC
   LVVTR = LVVAL
   LVVAL = -100
   IF (LVVTR.NE.-1) GO TO
   CALL LVGRN(LV1     AAC)
   LVDEST=  0
   LVTYPE(1) =  0
   LVVALS(1) = LV1     AAC
   LVVNVL =  1
   LVFUNC =         HOL
   LVVARG=LV1     AAB
   CALL LVNSRT
   IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
   IF(LVVAL.LT.0) RETURN
      NTEMP = LV1     AAC
 4 CONTINUE                                                    PARSE 49
   IF(ERRFLG) GO TO 25                                         PARSE 50
   ST=IABS(STR(I))                                             PARSE 51
   IF(STR(I) .LT. 0) GO TO 6                                   PARSE 52
   LVDEST=  0
   LV1     AAB = ST
   LVTYPE(1) =  1
   LVVALS(1) = LV1     AAB
   LVDEST=  0
   LVVNVL =  1
   LVFUNC =         HOL
   LVVARG=      NTEMP
   CALL LVNSRT
   IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
   IF(LVVAL.LT.0) RETURN
   IF(ERRFLG) GO TO 25                                         PARSE 54
   LVDEST=  0
   LVTYPE(1) =  0
   LVVALS(1) =      OPRAND
   LVVNVL =  1
   LVFUNC =     STRING
   LVVARG=     STRING
   CALL LVNSRT
   IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
   IF(LVVAL.LT.0) RETURN
   LVVTR = LVVAL
   LVVAL = -100
   IF (LVVTR.NE.-1) GO TO          10
   IF(ERRFLG) GO TO 25                                         PARSE 56
 6 STEMP=DICT(ST)                                              PARSE 57
   LVDEST=  0
   LV1     AAC = STEMP
   LVTYPE(1) =  1
   LVVALS(1) = LV1     AAC
   LVDEST=  0
   LVVNVL =  1
   LVFUNC =     STRING
   LVVARG=     STRING
   CALL LVNSRT
   IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
   IF(LVVAL.LT.0) RETURN
   IF(ERRFLG) GO TO 25                                         PARSE 59
```

```
 10 CONTINUE                                                          PARSE 60
    LV1    AAD =      OPRAND
    LVDEST=  0
    LV1    AAE = 0
    LVTYPE(1) =  1
    LVVALS(1) = LV1    AAE
    LVDEST=  0
    LVVNVL =  1
    LVFUNC =       FUNC2
    LVVARG=LV1     AAD
    CALL LVNSRT
    IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
    IF(LVVAL.LT.0) RETURN
    LVDEST=  0
    LV1    AAF = 0
    LVTYPE(1) =  1
    LVVALS(1) = LV1    AAF
    LVDEST=  0
    LVVNVL =  1
    LVFUNC =       FUNC3
    LVVARG=LV1     AAD
    CALL LVNSRT
    IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
    IF(LVVAL.LT.0) RETURN
    LVDEST=  0
    LV1    AAG = 0
    LVTYPE(1) =  1
    LVVALS(1) = LV1    AAG
    LVDEST=  0
    LVVNVL =  1
    LVFUNC =       LEVEL
    LVVARG=LV1     AAD
    CALL LVNSRT
    IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
    IF(LVVAL.LT.0) RETURN
    CALL RECOG(FAIL)                                                  PARSE 62
    IF(FAIL) GO TO 40                                                 PARSE 63
 25 CONTINUE                                                          PARSE 64
    IF(ERRFLG) PRINT 100                                              PARSE 65
100 FORMAT(/)                                                         PARSE 66
    CALL PRNTS                                                        PARSE 67
    NCHAR=0                                                           PARSE 68
 30 NCHAR=NCHAR+1                                                     PARSE 69
    LVVPOS =       NCHAR
    LVVTYP =    3
    LVFUNC=         HOL
    LVVARG=     STRING
    CALL LVFIND(LV2      E,LV2      F,LV2      G,LV2      H)
    LV1    AAD =      STRING
    IF (LVVAL.NE.-1) LV1    AAD = LVVAL
    LVVTR = LVVAL
    LVVAL = -100
    IF (LVVTR.EQ.-1) GO TO          35
    LV1    AAH = LV1    AAD
    LVVAD=-1
    LVVTYP=-1
    LVVPOS=1
```

```
      LVFUNC=       LEFT
      LVVARG=LV1     AAH
      CALL LVDLET
      LV1     AAH = LV1     AAD
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=      RIGHT
      LVVARG=LV1     AAH
      CALL LVDLET
      LV1     AAH = LV1     AAD
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=        HOL
      LVVARG=LV1     AAH
      CALL LVDLET
      LV1     AAH = LV1     AAD
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=     STRING
      LVVARG=LV1     AAH
      CALL LVDLET
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.EQ.-1) GO TO          30
      IF (LVVTR.NE.-1) GO TO          30
   35 CONTINUE                                                  PARSE 71
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=     STRING
      LVVARG=     STRING
      CALL LVDLET
      LV1     AAD =      OPRAND
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=     OPRAND
      LVVARG=LV1     AAD
      CALL LVDLET
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=     STRING
      LVVARG=LV1     AAD
      CALL LVDLET
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=     ACTION
      LVVARG=LV1     AAD
      CALL LVDLET
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
```

```
      LVFUNC=     FUNC1
      LVVARG=LV1    AAD
      CALL LVDLET
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=     FUNC2
      LVVARG=LV1    AAD
      CALL LVDLET
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=     FUNC3
      LVVARG=LV1    AAD
      CALL LVDLET
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=     LEVEL
      LVVARG=LV1    AAD
      CALL LVDLET
      NSTR=NCHRP                                                    PARSE 74
      RETURN                                                        PARSE 75
   40 PRINT 300,MAXJ                                                PARSE 76
  300 FORMAT(1X,34H PARSE FAILED AFTER CHARACTER NO. ,I3)           PARSE 77
      ERRFLG=.TRUE.                                                 PARSE 78
      GO TO 25                                                      PARSE 79
      RETURN
25000 CONTINUE
      LV2      A=LV2      B=LV2      C=LV2      D=0
      LV2      E=LV2      F=LV2      G=LV2      H=0
      GO TO 25001
      END
```

GIRL Version

```
$       SUBROUTINE PARSE                                                    PARSE     2
        COMMON/NEEDS/STJ,JSTACK,R,JAS,J,JLAST,RTEMP,STACK(400)              PARSE     3
        COMMON/FUNC/  NARY(5,12),MARGS,IARGS(50),FNCLOC(5),NFUNC            CY588     1
        COMMON /STRING/ NTYPE,NSTR,STR                                      PARSE     5
        COMMON /GIRL/NTERMS,PLUS,MINUS,SLASH,LPAR,RPAR,COMMA,STAR,EXP,LT,   PARSE     6
       +LE,GT,GE,EQ,NE,OR,AND,NOT,EQUALS,OPRAND                             PARSE     7
        COMMON /HL/ HOL,ACTION,FUNC1,FUNC2,FUNC3,LEFT,RIGHT,STRING,MAXJ     PARSE     8
        COMMON /NEED/ START,ASSOC,LEVEL,STOP                                PARSE     9
        COMMON /TYP/ NARRAY,TYPE1,TYPE2,ERRFLG                              PARSE    10
        COMMON/NOPAR/NOPAR,NDEP,NDEPTH,NFLAG                                PARSE    11
        COMMON /NTIMES/ NTIMES,I                                            PARSE    12
        COMMON/VAR/VFOR(15),NUMCHR,NCHRP,CHAR,NDICT                         CY588     2
        INTEGER TYPE1,TYPE2,START,TYP(3)                                    PARSE    14
        LOGICAL ERRFLG,FAIL                                                 PARSE    15
        INTEGER STR(1),STEMP,ST,DICT(19)                                    PARSE    16
        EQUIVALENCE(DICT(1),PLUS)                                           PARSE    17
        INTEGER PLUS,MINUS,SLASH,LPAR,RPAR,COMMA,STAR,EXP,LT,LE,GT,GE,EQ,   PARSE    18
       +NE,OR,AND,NOT,EQUALS,OPRAND,ASSOC,LEVEL,STOP,ACTION,HOL,LEFT,RIGHT  PARSE    19
       +,STRING,FUNC1,FUNC2,FUNC3                                           PARSE    20
        DATA NTIMES /0/                                                     PARSE    22
        IF(NTIMES .GT. 0) GO TO 3                                           PARSE    23
        NTIMES=1                                                            PARSE    24
G       EXECUTE                                                             PARSE    25
        CALL PHONEY                                                         PARSE    26
        CALL LVFECH(19)                                                     PARSE    27
        READ(19)PLUS,MINUS,SLASH,LPAR,RPAR,COMMA,STAR,EXP,LT,LE,GT,GE,EQ,   PARSE    28
       +NE,OR,AND,NOT,EQUALS,OPRAND,ASSOC,LEVEL,STOP,ACTION,HOL,LEFT,RIGHT  PARSE    29
       +,STRING,FUNC1,FUNC2,FUNC3,NTERMS,(TYP(I),I=1,3)                     PARSE    30
      3 IF(NSTR .LE. 0) RETURN                                              PARSE    31
        ERRFLG=.FALSE.                                                      PARSE    32
        START=TYP(NTYPE)                                                    PARSE    33
        NARGS=0                                                             PARSE    34
        NFLAG=0                                                             PARSE    35
        MAXJ=0                                                              PARSE    36
        NOPAR=0                                                             PARSE    37
        TYPE1=-1                                                            PARSE    38
        TYPE2=-1                                                            PARSE    39
        NARRAY=-1                                                           PARSE    40
        NDEPTH=0                                                            PARSE    41
        NDEP=0                                                              PARSE    42
        DO 20 I=1,50                                                        PARSE    43
     20 IARGS(I)=0                                                          PARSE    44
        DO 22 I=1,60                                                        PARSE    45
     22 NARY(I)=0                                                           PARSE    46
        DO 10 I=1,NSTR                                                      PARSE    47
G       STRING(+HOL.I 'NTEMP//4,HOL $ 'NTEMP)                               PARSE    48
      4 CONTINUE                                                            PARSE    49
        IF(ERRFLG) GO TO 25                                                 PARSE    50
        ST=IABS(STR(I))                                                     PARSE    51
        IF(STR(I) .LT. 0) GO TO 6                                           PARSE    52
G       NTEMP HOL **ST**                                                    PARSE    53
        IF(ERRFLG) GO TO 25                                                 PARSE    54
G       STRING STRING OPRAND//10                                            PARSE    55
        IF(ERRFLG) GO TO 25                                                 PARSE    56
      6 STEMP=DICT(ST)                                                      PARSE    57
G       STRING STRING **STEMP**                                             PARSE    58
        IF(ERRFLG) GO TO 25                                                 PARSE    59
```

```
   10 CONTINUE                                                          PARSE      60
G     OPRAND(FUNC2 ''0'',FUNC3 ''0'',LEVEL ''0'')                       PARSE      61
      CALL RECOG(FAIL)                                                 PARSE      62
      IF(FAIL) GO TO 40                                                PARSE      63
   25 CONTINUE                                                         PARSE      64
      IF(ERRFLG) PRINT 100                                             PARSE      65
  100 FORMAT(/)                                                        PARSE      66
      CALL PRNTS                                                       PARSE      67
      NCHAR=0                                                          PARSE      68
   30 NCHAR=NCHAR+1                                                    PARSE      69
G     STRING+HOL.NCHAR/35(-LEFT,-RIGHT,-HOL,-STRING/30/30)             PARSE      70
   35 CONTINUE                                                         PARSE      71
G     STRING-STRING                                                    PARSE      72
G     OPRAND(-OPRAND,-STRING,-ACTION,-FUNC1,-FUNC2,-FUNC3,-LEVEL)      PARSE      73
      NSTR=NCHRP                                                       PARSE      74
      RETURN                                                           PARSE      75
   40 PRINT 300,MAXJ                                                   PARSE      76
  300 FORMAT(1X,34H PARSE FAILED AFTER CHARACTER NO. ,I3)              PARSE      77
      ERRFLG=.TRUE.                                                    PARSE      78
      GO TO 25                                                         PARSE      79
G     COMPLETE                                                         PARSE      80
```

FORTRAN Version

```
      SUBROUTINE PHONEY                                                PHONEY 2
      INTEGER FLGSPC,FLAGSP
      COMMON /LVVTR1/LVVSZE,LVVGSP,NODSPC( 1000)/LVVTR2/LSTSPC( 1000)/
     *LVVTR3/LNKSPC( 1000)/LVVTR4/FLGSPC( 1000)
      COMMON/LVVTR5/LVFILE,LVCMPR,NODESP(     1)
     +/LVVTR6/LISTSP(     1)/LVVTR7/LINKSP(     1)
     +/LVVTR8/FLAGSP(     1)
      COMMON/LVRAND/LVKPRM,LVKS,LVKX,LVKOY,LVKOX,LVTEMP
      COMMON/LVARGS/LVFUNC,LVVARG,LVVAD,LVVPOS,LVVTYP,       LVVAL,
     +LVHEAD,LVVNVL,LVDEST,LVVALS(10),LVTYPE(10),LVSKIP
      COMMON/LVTABL/LVTSIZ,LVMAP(     1)/LVVSEQ/LVSIZE,LVSQSP(     1)
      LVVSZE =   1000
      LVFILE=  0
      LVCMPR= 0
      LVSIZE=     1
      LVSKIP= 1
      LVKPRM=  17
      CALL LVSETP
      GO TO 25000
25001 CONTINUE
      RETURN                                                           PHONEY 4
      STOP
25000 CONTINUE
      GO TO 25001
      END
```

GIRL Version

```
      SUBROUTINE PHONEY                                                PHONEY     2
G     EXECUTE                                                          PHONEY     3
      RETURN                                                           PHONEY     4
G     COMPLETE                                                         PHONEY     5
```

FORTRAN Version

```
      SUBROUTINE PRNTS                                                    PRNTS  2
      COMMON/LVARGS/LVFUNC,LVVARG,LVVAD,LVVPOS,LVVTYP,       LVVAL,
     +LVHEAD,LVVNVL,LVDEST,LVVALS(10),LVTYPE(10),LVSKIP
      COMMON/LVTABL/LVTSIZ,LVMAP(    1)/LVVSEQ/LVSIZE,LVSQSP(     1)
      COMMON /HL/ HOL,ACTION,FUNC1,FUNC2,FUNC3,LEFT,RIGHT,STRING          PRNTS  3
      COMMON /VAR/ VFOR,NCHAR,NCHARP,CHAR,NDICT                           PRNTS  4
      COMMON /TYP/ NARRAY,TYPE1,TYPE2,ERRFLG                              PRNTS  5
      COMMON /STRING/ NTYPE,NSTR,STR                                      PRNTS  6
      COMMON /GIRL/ NNN(19),OPRAND                                        PRNTS  7
      LOGICAL ERRFLG                                                      PRNTS  8
      INTEGER VFOR(15),CHAR,STRING,HOL,RIGHT,STR(1),OPRAND                PRNTS  9
      GO TO 25000
25001 CONTINUE
      NCHAR=0                                                             PRNTS 11
      NCHARP=0                                                            PRNTS 12
      DO 5 I=1,15                                                         PRNTS 13
    5 VFOR(I)=0                                                           PRNTS 14
      NINT=NTMP=1                                                         PRNTS 15
      DO 10 I=1,NSTR                                                      PRNTS 16
      LV1    AAD =      STRING
      LVVPOS =           I
      LVVTYP =    3
      LVFUNC=         HOL
      LVVARG= LV1    AAD
      CALL LVFIND(LV2       A,LV2       B,LV2       C,LV2       D)
      LV1    AAI = LV1    AAD
      IF (LVVAL.NE.-1) LV1    AAI = LVVAL
            NODE = LV1    AAI
      LVVPOS =           I
      LVVTYP =    3
      LVFUNC=      STRING
      LVVARG= LV1    AAD
      CALL LVFIND(LV2       E,LV2       F,LV2       G,LV2       H)
      LV1    AAI = LV1    AAD
      IF (LVVAL.NE.-1) LV1    AAI = LVVAL
              N1 = LV1    AAI
      LVVAL = -100
      IF (        N1.NE.    OPRAND) LVVAL = -1
      LVVTP = LVVAL
      LVVAL = -100
      IF (LVVTP.EQ.-1) GO TO          16
      NINT=NINT+1                                                         PRNTS 18
   16 J=0                                                                 PRNTS 19
   20 J=J+1                                                               PRNTS 20
      LVVPOS =           J
      LVVTYP =    3
      LVFUNC=        LEFT
      LVVARG=        NODE
      CALL LVFIND(LV2       I,LV2       J,LV2       K,LV2       L)
      LV1    AAD =       NODE
      IF (LVVAL.NE.-1) LV1    AAD = LVVAL
      LVVTP = LVVAL
      LVVAL = -100
      IF (LVVTP.EQ.-1) GO TO          30
      LV1    AAI = LV1    AAD
      LVVPOS =           1
      LVVTYP =    3
```

```
      LVFUNC=         HOL
      LVVARG= LV1     AAI
      CALL LVFIND(LV2       M,LV2       N,LV2       O,LV2       P)
      LV1     AAJ = LV1     AAI
      IF (LVVAL.NE.-1) LV1     AAJ = LVVAL
           CHAR = LV1     AAJ
      LV1     AAJ = LV1     AAO
      LVVPOS =           2
      LVVTYP =   3
      LVFUNC=         HOL
      LVVARG= LV1     AAJ
      CALL LVFIND(LV2       Q,LV2       R,LV2       S,LV2       T)
      LV1     AAI = LV1     AAJ
      IF (LVVAL.NE.-1) LV1     AAI = LVVAL
           NDICT = LV1     AAI
      CALL FORM                                                      PRNTS 22
      GO TO 20                                                       PRNTS 23
   30 CONTINUE                                                       PRNTS 24
      IF(NINT .GT. NTMP) GO TO 35                                    PRNTS 25
      LV1     AAO =          N1
      LVVPOS =           1
      LVVTYP =   3
      LVFUNC=         HOL
      LVVARG= LV1     AAO
      CALL LVFIND(LV2       U,LV2       V,LV2       W,LV2       X)
      LV1     AAI = LV1     AAO
      IF (LVVAL.NE.-1) LV1     AAI = LVVAL
           CHAR = LV1     AAI
      LVVPOS =           2
      LVVTYP =   3
      LVFUNC=         HOL
      LVVARG= LV1     AAO
      CALL LVFIND(LV2       Y,LV2       Z,LV2       0,LV2       1)
      LV1     AAI = LV1     AAO
      IF (LVVAL.NE.-1) LV1     AAI = LVVAL
           NDICT = LV1     AAI
      CALL FORM                                                      PRNTS 27
      GO TO 37                                                       PRNTS 28
   35 NTMP=NINT                                                      PRNTS 29
      LVVTYP = 3
      LVVPOS = 1
      LVINDX = 0
      LVFUNC=         HOL
      LVVARG=        NODE
      CALL LVFIND(LVINDX,LVINDX,LVINDX,LVINDX)
      LV1     AAO =         NODE
      IF (LVVAL.NE.-1) LV1     AAO = LVVAL
           NDICT = LV1     AAO
      LVVTYP = 3
      LVVPOS = 1
      LVINDX = 0
      LVFUNC=         HOL
      LVVARG=     OPRAND
      CALL LVFIND(LVINDX,LVINDX,LVINDX,LVINDX)
      LV1     AAO =      OPRAND
      IF (LVVAL.NE.-1) LV1     AAO = LVVAL
           CHAR = LV1     AAO
```

```
      CALL FORM                                                        PRNTS 32
   37 J=0                                                              PRNTS 33
   40 J=J+1                                                            PRNTS 34
      LVVPOS =           J
      LVVTYP =   3
      LVFUNC=       RIGHT
      LVVARG=        NODE
      CALL LVFIND(LV2       2,LV2       3,LV2       4,LV2       5)
      LV1    AAD =        NODE
      IF (LVVAL.NE.-1) LV1    AAD = LVVAL
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.EQ.-1) GO TO          10
      LV1    AAI = LV1    AAD
      LVVPOS =           1
      LVVTYP =   3
      LVFUNC=         HOL
      LVVARG= LV1    AAI
      CALL LVFIND(LV2       6,LV2       7,LV2       8,LV2       9)
      LV1    AAJ = LV1    AAI
      IF (LVVAL.NE.-1) LV1    AAJ = LVVAL
           CHAR = LV1    AAJ
      LV1    AAJ = LV1    AAD
      LVVPOS =           2
      LVVTYP =   3
      LVFUNC=         HOL
      LVVARG= LV1    AAJ
      CALL LVFIND(LV2      AA,LV2      AB,LV2      AC,LV2      AD)
      LV1    AAI = LV1    AAJ
      IF (LVVAL.NE.-1) LV1    AAI = LVVAL
           NDICT = LV1    AAI
      CALL FORM                                                        PRNTS 36
      GO TO 40                                                         PRNTS 37
   10 CONTINUE                                                         PRNTS 38
      NC=1+(NCHARP-1)/8                                                PRNTS 39
  100 FORMAT(1X,15A8)                                                  PRNTS 40
      IF(ERRFLG) PRINT 100,(VFOR(I),I=1,NC)                            PRNTS 41
      RETURN
25000 CONTINUE
      LV2       A=LV2       B=LV2       C=LV2       D=0
      LV2       E=LV2       F=LV2       G=LV2       H=0
      LV2       I=LV2       J=LV2       K=LV2       L=0
      LV2       M=LV2       N=LV2       O=LV2       P=0
      LV2       Q=LV2       R=LV2       S=LV2       T=0
      LV2       U=LV2       V=LV2       W=LV2       X=0
      LV2       Y=LV2       Z=LV2       0=LV2       1=0
      LV2       2=LV2       3=LV2       4=LV2       5=0
      LV2       6=LV2       7=LV2       8=LV2       9=0
      LV2      AA=LV2      AB=LV2      AC=LV2      AD=0
      GO TO 25001
      END
```

GIRL Version

```
$     SUBROUTINE PRNTS                                                    PRNTS     2
      COMMON /HL/ HOL,ACTION,FUNC1,FUNC2,FUNC3,LEFT,RIGHT,STRING          PRNTS     3
      COMMON /VAR/ VFOR,NCHAR,NCHARP,CHAR,NDICT                           PRNTS     4
      COMMON /TYP/ NARRAY,TYPE1,TYPE2,ERRFLG                              PRNTS     5
      COMMON /STRING/ NTYPE,NSTR,STR                                      PRNTS     6
      COMMON /GIRL/ NNN(19),OPRAND                                        PRNTS     7
      LOGICAL ERRFLG                                                      PRNTS     8
      INTEGER VFOR(15),CHAR,STRING,HOL,RIGHT,STR(1),OPRAND                PRNTS     9
G     EXECUTE                                                             PRNTS    10
      NCHAR=0                                                             PRNTS    11
      NCHARP=0                                                            PRNTS    12
      DO 5 I=1,15                                                         PRNTS    13
    5 VFOR(I)=0                                                           PRNTS    14
      NINT=NTMP=1                                                         PRNTS    15
      DO 10 I=1,NSTR                                                      PRNTS    16
G     STRING(+HOL.I *NODE,+STRING.I *N1=OPRAND/16)                        PRNTS    17
      NINT=NINT+1                                                         PRNTS    18
   16 J=0                                                                 PRNTS    19
   20 J=J+1                                                               PRNTS    20
G     NODE+LEFT.J/30 (+HOL.1 *CHAR,+HOL.2 *NDICT)                         PRNTS    21
      CALL FORM                                                           PRNTS    22
      GO TO 20                                                            PRNTS    23
   30 CONTINUE                                                            PRNTS    24
      IF(NINT .GT. NTMP) GO TO 35                                         PRNTS    25
G     N1(+HOL.1 *CHAR,+HOL.2 *NDICT)                                      PRNTS    26
      CALL FORM                                                           PRNTS    27
      GO TO 37                                                            PRNTS    28
   35 NTMP=NINT                                                           PRNTS    29
G     NODE+HOL *NDICT                                                     PRNTS    30
G     OPRAND+HOL *CHAR                                                    PRNTS    31
      CALL FORM                                                           PRNTS    32
   37 J=0                                                                 PRNTS    33
   40 J=J+1                                                               PRNTS    34
G     NODE+RIGHT.J/10 (+HOL.1 *CHAR,+HOL.2 *NDICT)                        PRNTS    35
      CALL FORM                                                           PRNTS    36
      GO TO 40                                                            PRNTS    37
   10 CONTINUE                                                            PRNTS    38
      NC=1+(NCHARP-1)/8                                                   PRNTS    39
  100 FORMAT(1X,15A8)                                                     PRNTS    40
      IF(ERRFLG) PRINT 100,(VFOR(I),I=1,NC)                               PRNTS    41
G     COMPLETE                                                            PRNTS    42
```

```
      SUBROUTINE PROG                                                     PROG      2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),    RICH      2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,          CY58A    80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                    RICH      4
      DIMENSION IALPH(7)                                                  PROG      4
      DATA (IALPH(I),I=1,7)/1HP,1HR,1HO,1HG,1HR,1HA,1HM/                  PROG      5
C** PROGRAM STATEMENT PROCESSOR                                           PROG      6
      JPTR=7                                                              PROG      7
C** CHECK SPELLING                                                        PROG      8
      DO 5 I=1,7                                                          PROG      9
      IF(NEXT(JPTR) .NE. IALPH(I)) GO TO 10                               PROG     10
    5 CONTINUE                                                            PROG     11
C** GET PROGRAM NAME AND STORE IN SYMBOL TABLE                            PROG     12
      CALL GNLE                                                           PROG     13
      IF(JTYP .NE. 2) GO TO 10                                            PROG     14
      IDTYP=2                                                             PROG     15
      CALL STORE                                                          PROG     16
      RETURN                                                              PROG     17
   10 CALL ERROR(7)                                                       PROG     18
      RETURN                                                              PROG     19
      END                                                                 PROG     20
```

40 Bits

```
REAL FUNCTION Q1REAL(X)
DATA R/7777777777777740000008/
Q1REAL=AND(X,R)
RETURN
END
DOUBLE PRECISION FUNCTION Q1DPRE(X)
DOUBLE PRECISION X,T
DIMENSION W(2)
EQUIVALENCE(T,W(1))
DATA R/7777777777777740000008/
T=X
W(1)=AND(W(1),R)
W(2)=AND(W(2),R)
Q1DPRE=T
RETURN
END
COMPLEX FUNCTION Q1COMP(X)
COMPLEX X,T
DIMENSION W(2)
EQUIVALENCE(T,W(1))
DATA R/7777777777777740000008/
T=X
W(1)=AND(W(1),R)
W(2)=AND(W(2),R)
Q1COMP=T
RETURN
END
```

39 Bits

```
REAL FUNCTION Q1REAL(X)
DATA R/7777777777777700000008/
Q1REAL=AND(X,R)
RETURN
END
DOUBLE PRECISION FUNCTION Q1DPRE(X)
DOUBLE PRECISION X,T
DIMENSION W(2)
EQUIVALENCE(T,W(1))
DATA R/7777777777777700000008/
T=X
W(1)=AND(W(1),R)
W(2)=AND(W(2),R)
Q1DPRE=T
RETURN
END
COMPLEX FUNCTION Q1COMP(X)
COMPLEX X,T
DIMENSION W(2)
EQUIVALENCE(T,W(1))
DATA R/7777777777777700000008/
T=X
W(1)=AND(W(1),R)
W(2)=AND(W(2),R)
Q1COMP=T
RETURN
END
```

38 Bits

```
REAL FUNCTION Q1REAL(X)
DATA R/77777777777600000008/
Q1REAL=AND(X,R)
RETURN
END
DOUBLE PRECISION FUNCTION Q1DPRE(X)
DOUBLE PRECISION X,T
DIMENSION W(2)
EQUIVALENCE(T,W(1))
DATA R/77777777777600000008/
T=X
W(1)=AND(W(1),R)
W(2)=AND(W(2),R)
Q1DPRE=T
RETURN
END
COMPLEX FUNCTION Q1COMP(X)
COMPLEX X,T
DIMENSION W(2)
EQUIVALENCE(T,W(1))
DATA R/77777777777600000008/
T=X
W(1)=AND(W(1),R)
W(2)=AND(W(2),R)
Q1COMP=T
RETURN
END
```

37 Bits

```
REAL FUNCTION Q1REAL(X)
DATA R/77777777777400000008/
Q1REAL=AND(X,R)
RETURN
END
DOUBLE PRECISION FUNCTION Q1DPRE(X)
DOUBLE PRECISION X,T
DIMENSION W(2)
EQUIVALENCE(T,W(1))
DATA R/77777777777400000008/
T=X
W(1)=AND(W(1),R)
W(2)=AND(W(2),R)
Q1DPRE=T
RETURN
END
COMPLEX FUNCTION Q1COMP(X)
COMPLEX X,T
DIMENSION W(2)
EQUIVALENCE(T,W(1))
DATA R/77777777777400000008/
T=X
W(1)=AND(W(1),R)
W(2)=AND(W(2),R)
Q1COMP=T
RETURN
END
```

36 Bits

```
REAL FUNCTION Q1REAL(X)
DATA R/777777777777000000008/
Q1REAL=AND(X,R)
RETURN
END
DOUBLE PRECISION FUNCTION Q1DPRE(X)
DOUBLE PRECISION X,T
DIMENSION W(2)
EQUIVALENCE(T,W(1))
DATA R/7777777777770000000008/
T=X
W(1)=AND(W(1),R)
W(2)=AND(W(2),R)
Q1DPRE=T
RETURN
END
COMPLEX FUNCTION Q1COMP(X)
COMPLEX X,T
DIMENSION W(2)
EQUIVALENCE(T,W(1))
DATA R/7777777777770000000008/
T=X
W(1)=AND(W(1),R)
W(2)=AND(W(2),R)
Q1COMP=T
RETURN
END
```

35 Bits

```
REAL FUNCTION Q1REAL(X)
DATA R/777777777776000000008/
Q1REAL=AND(X,R)
RETURN
END
DOUBLE PRECISION FUNCTION Q1DPRE(X)
DOUBLE PRECISION X,T
DIMENSION W(2)
EQUIVALENCE(T,W(1))
DATA R/7777777777760000000008/
T=X
W(1)=AND(W(1),R)
W(2)=AND(W(2),R)
Q1DPRE=T
RETURN
END
COMPLEX FUNCTION Q1COMP(X)
COMPLEX X,T
DIMENSION W(2)
EQUIVALENCE(T,W(1))
DATA R/7777777777760000000008/
T=X
W(1)=AND(W(1),R)
W(2)=AND(W(2),R)
Q1COMP=T
RETURN
END
```

34 Bits

```
REAL FUNCTION Q1REAL(X)
DATA R/777777777774000000008/
Q1REAL=AND(X,R)
RETURN
END
DOUBLE PRECISION FUNCTION Q1DPRE(X)
DOUBLE PRECISION X,T
DIMENSION W(2)
EQUIVALENCE(T,W(1))
DATA R/777777777774000000008/
T=X
W(1)=AND(W(1),R)
W(2)=AND(W(2),R)
Q1DPRE=T
RETURN
END
COMPLEX FUNCTION Q1COMP(X)
COMPLEX X,T
DIMENSION W(2)
EQUIVALENCE(T,W(1))
DATA R/777777777774000000008/
T=X
W(1)=AND(W(1),R)
W(2)=AND(W(2),R)
Q1COMP=T
RETURN
END
```

33 Bits

```
REAL FUNCTION Q1REAL(X)
DATA R/777777777770000000008/
Q1REAL=AND(X,R)
RETURN
END
DOUBLE PRECISION FUNCTION Q1DPRE(X)
DOUBLE PRECISION X,T
DIMENSION W(2)
EQUIVALENCE(T,W(1))
DATA R/777777777770000000008/
T=X
W(1)=AND(W(1),R)
W(2)=AND(W(2),R)
Q1DPRE=T
RETURN
END
COMPLEX FUNCTION Q1COMP(X)
COMPLEX X,T
DIMENSION W(2)
EQUIVALENCE(T,W(1))
DATA R/777777777770000000008/
T=X
W(1)=AND(W(1),R)
W(2)=AND(W(2),R)
Q1COMP=T
RETURN
END
```

32 Bits

```
      REAL FUNCTION Q1REAL(X)
      DATA R/7777777777760000000008/
      Q1REAL=AND(X,R)
      RETURN
      END
      DOUBLE PRECISION FUNCTION Q1DPRE(X)
      DOUBLE PRECISION X,T
      DIMENSION W(2)
      EQUIVALENCE(T,W(1))
      DATA R/7777777777760000000008/
      T=X
      W(1)=AND(W(1),R)
      W(2)=AND(W(2),R)
      Q1DPRE=T
      RETURN
      END
      COMPLEX FUNCTION Q1COMP(X)
      COMPLEX X,T
      DIMENSION W(2)
      EQUIVALENCE(T,W(1))
      DATA R/7777777777760000000008/
      T=X
      W(1)=AND(W(1),R)
      W(2)=AND(W(2),R)
      Q1COMP=T
      RETURN
      END
```

31 Bits

```
      REAL FUNCTION Q1REAL(X)
      DATA R/7777777777400000000008/
      Q1REAL=AND(X,R)
      RETURN
      END
      DOUBLE PRECISION FUNCTION Q1DPRE(X)
      DOUBLE PRECISION X,T
      DIMENSION W(2)
      EQUIVALENCE(T,W(1))
      DATA R/7777777777400000000008/
      T=X
      W(1)=AND(W(1),R)
      W(2)=AND(W(2),R)
      Q1DPRE=T
      RETURN
      END
      COMPLEX FUNCTION Q1COMP(X)
      COMPLEX X,T
      DIMENSION W(2)
      EQUIVALENCE(T,W(1))
      DATA R/7777777777400000000008/
      T=X
      W(1)=AND(W(1),R)
      W(2)=AND(W(2),R)
      Q1COMP=T
      RETURN
      END
```

30 Bits

```
REAL FUNCTION Q1REAL(X)
DATA R/77777777770000000000B/
Q1REAL=AND(X,R)
RETURN
END
DOUBLE PRECISION FUNCTION Q1DPRE(X)
DOUBLE PRECISION X,T
DIMENSION W(2)
EQUIVALENCE(T,W(1))
DATA R/77777777770000000000B/
T=X
W(1)=AND(W(1),R)
W(2)=AND(W(2),R)
Q1DPRE=T
RETURN
END
COMPLEX FUNCTION Q1COMP(X)
COMPLEX X,T
DIMENSION W(2)
EQUIVALENCE(T,W(1))
DATA R/77777777770000000000B/
T=X
W(1)=AND(W(1),R)
W(2)=AND(W(2),R)
Q1COMP=T
RETURN
END
```

```
      SUBROUTINE REALCK                                                  REAL      2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),   RICH      2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,         CYSBA    80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                   RICH      4
      COMMON/LOGIC/LOG,LOGST                                             REAL      4
      COMMON/REALNO/IREAL,IRELND,IP                                      REAL      5
      INTEGER A,DECPT,EEE,PLUS,MINUS,DEE,ELOC                            REAL      6
      DATA DECPT/1H./,EEE/1HE/, PLUS/1H+/, MINUS/1H-/,DEE/1HD/           REAL      7
C** THIS ROUTINE CHECKS A CHARACTER STRING TO SEE IF IT CONSTITUTES      REAL      8
C** A REAL NUMBER                                                       REAL      9
      IDES=0                                                             REAL     10
      IRELND=0                                                           REAL     11
      IF(IP .GE. N) GO TO 90                                             REAL     12
C** CHECK THAT FIRST CHARACTER IS A DECIMAL POINT OR NUMBER             REAL     13
      IF(NEXT(IP) .EQ. DECPT) GO TO 5                                    REAL     14
      IF(ITYPE(IP) .EQ. 2) GO TO 10                                      REAL     15
      GO TO 90                                                           REAL     16
    5 IF(JPTR .GT. N) GO TO 90                                           REAL     17
      IF(ITYPE(JPTR) .NE. 2) GO TO 90                                    REAL     18
      GO TO 20                                                           REAL     19
   10 IF(JPTR .GT. N) GO TO 90                                           REAL     20
   12 IF(ITYPE(JPTR) .EQ. 2) GO TO 15                                    REAL     21
C** CHECK THAT STRING CONTAINS A DECIMAL POINT                          REAL     22
      IF(A(JPTR-1) .NE. DECPT) GO TO 90                                  REAL     23
      LOGST=JPTR                                                         REAL     24
      CALL LOGCHK                                                        REAL     25
      IF(LOG .EQ. 1) GO TO 90                                            REAL     26
      JPTR=LOGST                                                         REAL     27
      GO TO 20                                                           REAL     28
   15 IF(JPTR .GT. N) GO TO 90                                           REAL     29
      GO TO 12                                                           REAL     30
   20 IREAL=1                                                            REAL     31
      IF(JPTR .GT. N) GO TO 35                                           REAL     32
   22 IF(ITYPE(JPTR) .EQ. 2) GO TO 25                                    REAL     33
      IF(A(JPTR-1) .EQ. EEE) GO TO 24                                    REAL     34
      IF(A(JPTR-1) .NE. DEE) GO TO 30                                    REAL     35
      IDES=1                                                             REAL     36
   24 ELOC=JPTR-2                                                        REAL     37
      GO TO 40                                                           REAL     38
   25 IF(JPTR .GT. N) GO TO 35                                           REAL     39
      GO TO 22                                                           REAL     40
   30 ELOC=JPTR-2                                                        REAL     41
   32 IRELND=ELOC                                                        REAL     42
C** NUMBER IS NOT IN "E" OR "D" FORMAT, RETURN                          REAL     43
      RETURN                                                             REAL     44
   35 IRELND=N                                                           REAL     45
      RETURN                                                             REAL     46
   40 IF(JPTR .GT. N) GO TO 32                                           REAL     47
C** NUMBER IS "E" OR "D" FORMAT                                         REAL     48
      NXT=NEXT(JPTR)                                                     REAL     49
      IF(NXT .EQ. PLUS .OR. NXT .EQ. MINUS) GO TO 45                     REAL     50
      IF(ITYPE(JPTR-1) .NE. 2) GO TO 32                                  REAL     51
      IF(JPTR .GT. N) GO TO 35                                           REAL     52
      GO TO 47                                                           REAL     53
   45 IF(JPTR .GT. N) GO TO 32                                           REAL     54
      IF(ITYPE(JPTR) .NE. 2) GO TO 32                                    REAL     55
   47 IF(ITYPE(JPTR) .EQ. 2) GO TO 50                                    REAL     56
      IRELND=JPTR-2                                                      REAL     57
      RETURN                                                             REAL     58
   50 IF(JPTR .GT. N) GO TO 35                                           REAL     59
      GO TO 47                                                           REAL     60
   90 IREAL=0                                                            REAL     61
      RETURN                                                             REAL     62
      END                                                                REAL     63
```

FORTRAN Version

```
      SUBROUTINE RECOG(FIN)                                                  RECOG  2
      COMMON/LVARGS/LVFUNC,LVVARG,LVVAD,LVVPOS,LVVTYP,        LVVAL,
     +LVHEAD,LVVNVL,LVDEST,LVVALS(10),LVTYPE(10),LVSKIP
      COMMON/LVTABL/LVTSIZ,LVMAP(     1)/LVVSEQ/LVSIZE,LVSQSP(     1)
      COMMON /HL/ HOL,ACTION,FUNC1,FUNC2,FUNC3,LEFT,RIGHT,STRING,MAXJ        RECOG  3
      COMMON /NEED/ START,ASSOC,LEVEL,STOP                                   RECOG  4
      COMMON /STRING/ NNN(2),STR                                             RECOG  5
      COMMON/NEEDS/STJ,JSTACK,R,JAS,J,JLAST,RTEMP,STACK(400)                 RECOG  6
      INTEGER HOL                                                            RECOG  7
      INTEGER START,ASSOC,STOP,RETRN,R,STJ,STACK,STR(1),ACTION,STRING,       RECOG  8
     $ RTEMP                                                                 RECOG  9
      LOGICAL FAIL,FIN                                                       RECOG 10
      FIN=.FALSE.                                                            RECOG 11
      GO TO 25000
25001 CONTINUE
      JSTACK=0                                                               RECOG 13
      J=1                                                                    RECOG 14
               R =       START
      LVVPOS =            J
      LVVTYP =    3
      LVFUNC=      STRING
      LVVARG=      STRING
      CALL LVFIND(LV2       A,LV2       B,LV2       C,LV2       D)
      LV1     AAD =      STRING
      IF (LVVAL.NE.-1) LV1     AAD = LVVAL
      LVVTP = LVVAL
      LVVAL = -100
      IF (LVVTP.EQ.-1) GO TO          70
            STJ = LV1     AAD
      M=-1                                                                   RECOG 17
      LVVPOS =            1
      LVVTYP =    3
      LVFUNC=         HOL
      LVVARG=      STRING
      CALL LVFIND(LV2       E,LV2       F,LV2       G,LV2       H)
      LV1     AAD =      STRING
      IF (LVVAL.NE.-1) LV1     AAD = LVVAL
      LVDEST=   0
      LV1     AAH = M
      LVTYPE(1) =  1
      LVVALS(1) = LV1     AAH
      LVDEST=   0
      LVVNVL =   1
      LVFUNC =      STRING
      LVVARG=LV1     AAD
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
    6 CONTINUE                                                               RECOG 19
   12 CONTINUE
      LV1     AAD =           R
      LVVTYP = 3
      LVVPOS = 1
      LVINDX = 0
      LVFUNC=       ASSOC
      LVVARG= LV1     AAD
      CALL LVFIND(LVINDX,LVINDX,LVINDX,LVINDX)
```

```
   LV1    AAI = LV1    AAD
   IF (LVVAL.NE.-1) LV1    AAI = LVVAL
   LVVTR = LVVAL
   LVVAL = -100
   IF (LVVTR.NE.-1) GO TO          15
   LVVTYP = 3
   LVVPOS = 1
   LVINDX = 0
   LVFUNC=       STOP
   LVVARG= LV1    AAD
   CALL LVFIND(LVINDX,LVINDX,LVINDX,LVINDX)
   LV1    AAI = LV1    AAD
   IF (LVVAL.NE.-1) LV1    AAI = LVVAL
   LVVTR = LVVAL
   LVVAL = -100
   IF (LVVTR.NE.-1) GO TO          15
   LVVAL = -100
   IF (LV1    AAD.NE.       STOP) LVVAL = -1
   LVVTR = LVVAL
   LVVAL = -100
   IF (LVVTR.EQ.-1) GO TO          20
15 JSTACK=JSTACK+1                                          RECOG 21
   STACK(JSTACK)=SHIFT(R,45) .OR. SHIFT(J,15)               RECOG 22
20 CONTINUE
   LVVTYP = 3
   LVVPOS = 1
   LVINDX = 0
   LVFUNC=     ACTION
   LVVARG=          R
   CALL LVFIND(LVINDX,LVINDX,LVINDX,LVINDX)
   LV1    AAD =          R
   IF (LVVAL.NE.-1) LV1    AAD = LVVAL
   LVVTR = LVVAL
   LVVAL = -100
   IF (LVVTR.EQ.-1) GO TO          22
         N = LV1    AAD
   CALL SEMANT(N,FAIL)                                      RECOG 24
   IF(FAIL) GO TO 99                                        RECOG 25
   GO TO 25                                                 RECOG 26
22 CONTINUE
   LVVTYP = 3
   LVVPOS = 1
   LVINDX = 0
   LVFUNC=        STJ
   LVVARG=          R
   CALL LVFIND(LVINDX,LVINDX,LVINDX,LVINDX)
   LV1    AAD =          R
   IF (LVVAL.NE.-1) LV1    AAD = LVVAL
   LVVTR = LVVAL
   LVVAL = -100
   IF (LVVTR.EQ.-1) GO TO          99
         R = LV1    AAD
25 J=J+1                                                    RECOG 28
   IF(J .GT. MAXJ) MAXJ=J                                   RECOG 29
31 CONTINUE
   LVVPOS =          J
   LVVTYP =   3
```

```
      LVFUNC=     STRING
      LVVARG=     STRING
      CALL LVFIND(LV2       I,LV2       J,LV2       K,LV2       L)
      LV1    AAD =      STRING
      IF (LVVAL.NE.-1) LV1     AAD = LVVAL
            STJ = LV1     AAD
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.NE.-1) GO TO           6
   40 STJ=-1                                                          RECOG 31
      LVVTYP = 3
      LVVPOS = 1
      LVINDX = 0
      LVFUNC=     ACTION
      LVVARG=           R
      CALL LVFIND(LVINDX,LVINDX,LVINDX,LVINDX)
      LV1    AAD =           R
      IF (LVVAL.NE.-1) LV1     AAD = LVVAL
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.EQ.-1) GO TO           42
               N = LV1     AAD
      CALL SEMANT(N,FAIL)                                             RECOG 33
      IF(FAIL) GO TO 99                                               RECOG 34
   42 CALL SSTOP(FAIL)                                                RECOG 35
      IF(FAIL) GO TO 99                                               RECOG 36
      JSTACK=JSTACK+1                                                 RECOG 37
      STACK(JSTACK)=SHIFT(R,45) .OR. SHIFT(J,15)                      RECOG 38
      LVVTYP = 3
      LVVPOS = 1
      LVINDX = 0
      LVFUNC=     ACTION
      LVVARG=           R
      CALL LVFIND(LVINDX,LVINDX,LVINDX,LVINDX)
      LV1    AAD =           R
      IF (LVVAL.NE.-1) LV1     AAD = LVVAL
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.EQ.-1) GO TO           44
               N = LV1     AAD
      CALL SEMANT(N,FAIL)                                             RECOG 40
      IF(FAIL) GO TO 99                                               RECOG 41
   44 CALL SLEVEL(FAIL)                                               RECOG 42
      IF(FAIL) RETURN                                                 RECOG 43
      GO TO 40                                                        RECOG 44
   99 CONTINUE                                                        RECOG 45
      CALL RECOV(RETRN)                                               RECOG 46
      IF(RETRN .LT. 0) GO TO 70                                       RECOG 47
      LVVAL = -100
      IF (     RETRN.NE.     ASSOC) LVVAL = -1
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.NE.-1) GO TO           30
      IF(RETRN .EQ. 0) GO TO 10                                       RECOG 49
      CALL SLEVEL(FAIL)                                               RECOG 50
      IF(FAIL) GO TO 65                                               RECOG 51
      GO TO 30                                                        RECOG 52
   65 IF(JSTACK .LE. 1) GO TO 70                                      RECOG 53
      JSTACK=JSTACK-1                                                 RECOG 54
      GO TO 99                                                        RECOG 55
   70 FIN=.TRUE.                                                      RECOG 56
      RETURN                                                          RECOG 57
      RETURN
25000 CONTINUE
      LV2       A=LV2       B=LV2       C=LV2       D=0
      LV2       E=LV2       F=LV2       G=LV2       H=0
      LV2       I=LV2       J=LV2       K=LV2       L=0
      GO TO 25001
      END
```

GIRL Version

```
$       SUBROUTINE RECOG(FIN)                                          RECOG     2
        COMMON /HL/ HOL,ACTION,FUNC1,FUNC2,FUNC3,LEFT,RIGHT,STRING,MAXJ RECOG     3
        COMMON /NEED/ START,ASSOC,LEVEL,STOP                           RECOG     4
        COMMON /STRING/ NNN(2),STR                                     RECOG     5
        COMMON/NEEDS/STJ,JSTACK,R,JAS,J,JLAST,RTEMP,STACK(400)         RECOG     6
        INTEGER HOL                                                    RECOG     7
        INTEGER START,ASSOC,STOP,RETRN,R,STJ,STACK,STR(1),ACTION,STRING, RECOG    8
      $ RTEMP                                                          RECOG     9
        LOGICAL FAIL,FIN                                               RECOG    10
        FIN=.FALSE.                                                    RECOG    11
G       EXECUTE                                                        RECOG    12
        JSTACK=0                                                       RECOG    13
        J=1                                                            RECOG    14
G       START 'R                                                       RECOG    15
G       STRING+STRING.J/70 'STJ                                        RECOG    16
        M=-1                                                           RECOG    17
G       STRING+HOL.1 STRING ''M''                                      RECOG    18
     6  CONTINUE                                                       RECOG    19
G   10  R(+ASSOC//15,+STOP//15,=STOP/20)                              RECOG    20
    15  JSTACK=JSTACK+1                                                RECOG    21
        STACK(JSTACK)=SHIFT(R,45) .OR. SHIFT(J,15)                     RECOG    22
G   20  R+ACTION/22 'N                                                 RECOG    23
        CALL SEMANT(N,FAIL)                                            RECOG    24
        IF(FAIL) GO TO 99                                              RECOG    25
        GO TO 25                                                       RECOG    26
G   22  R+STJ/99 'R                                                    RECOG    27
    25  J=J+1                                                          RECOG    28
        IF(J .GT. MAXJ) MAXJ=J                                         RECOG    29
G   30  STRING+STRING.J 'STJ//6                                       RECOG    30
    40  STJ=-1                                                         RECOG    31
G       R+ACTION/42 'N                                                 RECOG    32
        CALL SEMANT(N,FAIL)                                            RECOG    33
        IF(FAIL) GO TO 99                                              RECOG    34
    42  CALL SSTOP(FAIL)                                               RECOG    35
        IF(FAIL) GO TO 99                                              RECOG    36
        JSTACK=JSTACK+1                                                RECOG    37
        STACK(JSTACK)=SHIFT(R,45) .OR. SHIFT(J,15)                     RECOG    38
G       R+ACTION/44 'N                                                 RECOG    39
        CALL SEMANT(N,FAIL)                                            RECOG    40
        IF(FAIL) GO TO 99                                              RECOG    41
    44  CALL SLEVEL(FAIL)                                              RECOG    42
        IF(FAIL) RETURN                                                RECOG    43
        GO TO 40                                                       RECOG    44
    99  CONTINUE                                                       RECOG    45
        CALL RECOV(RETRN)                                              RECOG    46
        IF(RETRN .LT. 0) GO TO 70                                      RECOG    47
G       RETRN=ASSOC//30                                                RECOG    48
        IF(RETRN .EQ. 0) GO TO 10                                      RECOG    49
        CALL SLEVEL(FAIL)                                              RECOG    50
        IF(FAIL) GO TO 65                                              RECOG    51
        GO TO 30                                                       RECOG    52
    65  IF(JSTACK .LE. 1) GO TO 70                                     RECOG    53
        JSTACK=JSTACK-1                                                RECOG    54
        GO TO 99                                                       RECOG    55
    70  FIN=.TRUE.                                                     RECOG    56
        RETURN                                                         RECOG    57
G       COMPLETE                                                       RECOG    58
```

FORTRAN Version

```
      SUBROUTINE RECOV(RETRN)                                              RECOV  2
      COMMON/LVARGS/LVFUNC,LVVARG,LVVAD,LVVPOS,LVVTYP,       LVVAL,
     +LVHEAD,LVVNVL,LVDEST,LVVALS(10),LVTYPE(10),LVSKIP
      COMMON/LVTABL/LVTSIZ,LVMAP(    1)/LVVSEQ/LVSIZE,LVSOSP(    1)
      COMMON /NEED/ START,ASSOC,LEVEL,STOP                                 RECOV  3
      COMMON/NEEDS/STJ,JSTACK,R,JAS,J,JLAST,RTEMP,STACK(400)               RECOV  4
      COMMON /STRING/ NNN(2),STR                                           RECOV  5
      COMMON /HL/ HOL,ACTION,FUNC1,FUNC2,FUNC3,LEFT,RIGHT,STRING           RECOV  6
      INTEGER START,ASSOC,STOP,STACK,STR(1),STJ,R,STRING,RETRN,TEMP        RECOV  7
     $ ,RIGHT,HOL,BITGET,BITPUT                                            RECOV  8
      GO TO 25000
25001 CONTINUE
   10 R=BITGET(STACK(JSTACK),15,15)                                        RECOV 10
      JAS=BITGET(STACK(JSTACK),30,15)+1                                    RECOV 11
      LVVPOS =         JAS
      LVVTYP =    3
      LVFUNC=       ASSOC
      LVVARG=          R
      CALL LVFIND(LV2      A,LV2      B,LV2      C,LV2      D)
      LV1    AAD =          R
      IF (LVVAL.NE.-1) LV1    AAD = LVVAL
            TEMP = LV1    AAD
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.NE.-1) GO TO          30
   15 CONTINUE
      LV1    AAD =           R
      LVVAL = -100
      IF (LV1   . AAD.NE.       STOP) LVVAL = -1
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.NE.-1) GO TO          40
      LVVTYP = 3
      LVVPOS = 1
      LVINDX = 0
      LVFUNC=        STOP
      LVVARG= LV1    AAD
      CALL LVFIND(LVINDX,LVINDX,LVINDX,LVINDX)
      LV1    AAH = LV1    AAD
      IF (LVVAL.NE.-1) LV1    AAH = LVVAL
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.EQ.-1) GO TO          16
      LVVAL = -100
      IF (LV1    AAH.NE.       STOP) LVVAL = -1
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.NE.-1) GO TO          40
            R = LV1    AAH
      J=BITGET(STACK(JSTACK),45,15)                                        RECOV 14
      JSTACK=JSTACK-1                                                      RECOV 15
      RETRN=0                                                              RECOV 16
      RETURN                                                               RECOV 17
   16 JSTACK=JSTACK-1                                                      RECOV 18
      IF(JSTACK .LE. 0) GO TO 20                                           RECOV 19
      IF(BITGET(STACK(JSTACK),45,15) .LT. J) CALL SEMANT(0,FAIL)           RECOV 20
      J=BITGET(STACK(JSTACK),45,15)                                        RECOV 21
```

```
      GO TO 10                                                          RECOV 22
   20 RETRN=-1                                                          RECOV 23
      RETURN                                                            RECOV 24
   40 CONTINUE                                                          RECOV 25
      J=BITGET(STACK(JSTACK),45,15)                                     RECOV 26
      ISTCK=BITGET(STACK(JSTACK),60,15)                                 RECOV 27
      IF(ISTCK .GT. 0 .AND. ISTCK .LT. 777778) GO TO 16                 RECOV 28
      RETRN=STOP                                                        RECOV 29
      RETURN                                                            RECOV 30
   30 CONTINUE
               R =        TEMP
      IF(JSTACK .EQ. 1) GO TO 35                                        RECOV 32
      IF(R .NE. BITGET(STACK(JSTACK),15,15)) GO TO 35                   RECOV 33
      NTEMP=BITGET(STACK(JSTACK-1),15,15)                               RECOV 34
      JMARK=JSTACK                                                      RECOV 35
   31 STACK(JMARK)=STACK(JMARK) .OR. 777778                             RECOV 36
      JMARK=JMARK-1                                                     RECOV 37
      ISTCK=BITGET(STACK(JMARK),15,15)                                  RECOV 38
      IF(R .EQ. ISTCK .AND. BITGET(STACK(JMARK),60,15) .EQ. 777778)     RECOV 39
     $ GO TO 15                                                         RECOV 40
      IF(R .EQ. ISTCK .AND. JMARK .NE. 0) GO TO 31                      RECOV 41
      IF(R .NE. NTEMP .OR. JAS .NE. BITGET(STACK(JSTACK-1),30,15)       RECOV 42
     $ .OR. BITGET(STACK(JSTACK-1),60,15) .NE. 777778) GO TO 35         RECOV 43
      GO TO 15                                                          RECOV 44
   35 CONTINUE                                                          RECOV 45
      IF(BITGET(STACK(JSTACK),45,15) .LT. J) CALL SEMANT(0,FAIL)        RECOV 46
      STACK(JSTACK)=STACK(JSTACK) .AND. 777770000077777777778           RECOV 47
      STACK(JSTACK)=BITPUT(STACK(JSTACK),JAS,30)                        RECOV 48
      J=BITGET(STACK(JSTACK),45,15)                                     RECOV 49
      IF(BITGET(STACK(JSTACK),60,15) .NE. 777778)                       RECOV 50
     $ STACK(JSTACK)=STACK(JSTACK) .AND. 77777777777777000008           RECOV 51
      RETRN=ASSOC                                                       RECOV 52
      RETURN                                                            RECOV 53
      RETURN
25000 CONTINUE
      LV2       A=LV2       B=LV2       C=LV2       D=0
      GO TO 25001
      END
```

GIRL Version

```
$       SUBROUTINE RECOV(RETRN)                                          RECOV     2
        COMMON /NEED/ START,ASSOC,LEVEL,STOP                             RECOV     3
        COMMON/NEEDS/STJ,JSTACK,R,JAS,J,JLAST,RTEMP,STACK(400)           RECOV     4
        COMMON /STRING/ NNN(2),STR                                       RECOV     5
        COMMON /HL/ HOL,ACTION,FUNC1,FUNC2,FUNC3,LEFT,RIGHT,STRING       RECOV     6
        INTEGER START,ASSOC,STOP,STACK,STR(1),STJ,R,STRING,RETRN,TEMP     RECOV     7
       $ ,RIGHT,HOL,BITGET,BITPUT                                        RECOV     8
G       EXECUTE                                                          RECOV     9
   10   R=BITGET(STACK(JSTACK),15,15)                                    RECOV    10
        JAS=BITGET(STACK(JSTACK),30,15)+1                                RECOV    11
G       R+ASSOC.JAS 'TEMP//30                                            RECOV    12
G  15   R(=STOP//40,+STOP/16=STOP//40 'R)                                RECOV    13
        J=BITGET(STACK(JSTACK),45,15)                                    RECOV    14
        JSTACK=JSTACK-1                                                  RECOV    15
        RETRN=0                                                          RECOV    16
        RETURN                                                           RECOV    17
   16   JSTACK=JSTACK-1                                                  RECOV    18
        IF(JSTACK .LE. 0) GO TO 20                                       RECOV    19
        IF(BITGET(STACK(JSTACK),45,15) .LT. J) CALL SEMANT(0,FAIL)       RECOV    20
        J=BITGET(STACK(JSTACK),45,15)                                    RECOV    21
        GO TO 10                                                         RECOV    22
   20   RETRN=-1                                                         RECOV    23
        RETURN                                                           RECOV    24
   40   CONTINUE                                                         RECOV    25
        J=BITGET(STACK(JSTACK),45,15)                                    RECOV    26
        ISTCK=BITGET(STACK(JSTACK),60,15)                                RECOV    27
        IF(ISTCK .GT. 0 .AND. ISTCK .LT. 777778) GO TO 16                RECOV    28
        RETRN=STOP                                                       RECOV    29
        RETURN                                                           RECOV    30
G  30   TEMP 'R                                                          RECOV    31
        IF(JSTACK .EQ. 1) GO TO 35                                       RECOV    32
        IF(R .NE. BITGET(STACK(JSTACK),15,15)) GO TO 35                  RECOV    33
        NTEMP=BITGET(STACK(JSTACK-1),15,15)                              RECOV    34
        JMARK=JSTACK                                                     RECOV    35
   31   STACK(JMARK)=STACK(JMARK) .OR. 777778                            RECOV    36
        JMARK=JMARK-1                                                    RECOV    37
        ISTCK=BITGET(STACK(JMARK),15,15)                                 RECOV    38
        IF(R .EQ. ISTCK .AND. BITGET(STACK(JMARK),60,15) .EQ. 777778)    RECOV    39
      $ GO TO 15                                                         RECOV    40
        IF(R .EQ. ISTCK .AND. JMARK .NE. 0) GO TO 31                     RECOV    41
        IF(R .NE. NTEMP .OR. JAS .NE. BITGET(STACK(JSTACK-1),30,15)      RECOV    42
      $ .OR. BITGET(STACK(JSTACK-1),60,15) .NE. 777778) GO TO 35         RECOV    43
        GO TO 15                                                         RECOV    44
   35   CONTINUE                                                         RECOV    45
        IF(BITGET(STACK(JSTACK),45,15) .LT. J) CALL SEMANT(0,FAIL)       RECOV    46
        STACK(JSTACK)=STACK(JSTACK) .AND. 777770000077777777778          RECOV    47
        STACK(JSTACK)=BITPUT(STACK(JSTACK),JAS,30)                       RECOV    48
        J=BITGET(STACK(JSTACK),45,15)                                    RECOV    49
        IF(BITGET(STACK(JSTACK),60,15) .NE. 777778)                      RECOV    50
      $ STACK(JSTACK)=STACK(JSTACK) .AND. 777777777777777000008          RECOV    51
        RETRN=ASSOC                                                      RECOV    52
        RETURN                                                           RECOV    53
G       COMPLETE                                                         RECOV    54
```

```
      SUBROUTINE ROLCHK(I1,I2,I3,I4,I5,I6)
      DIMENSION IARG(6)
      DATA MASK/77000000000000000000B/
      IARG(1)=I1 .AND. MASK
      IARG(2)=I2 .AND. MASK
      IARG(3)=I3 .AND. MASK
      IARG(4)=I4 .AND. MASK
      IARG(5)=I5 .AND. MASK
      IARG(6)=I6 .AND. MASK
      ISUBNM=55555555B
      NSHIFT=66
      DO 10 I=1,6
      NSHIFT=NSHIFT-6
   10 ISUBNM=OR(ISUBNM,SHIFT(IARG(I),NSHIFT))
      WRITE(3) ISUBNM
      RETURN
      END
```

```
      SUBROUTINE SEARCH                                                  SEARCH     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),   RICH       2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,         CY58A     80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                   RICH       4
C** THIS ROUTINE SEARCHES THE SYMBOL TABLE FOR A NAME AND RETURNS        SEARCH     4
C**      ISRCH(1)=1 - NAME FOUND IN VARIABLE LIST      =0 - NOT FOUND     SEARCH     5
C**      ISRCH(2)=1 - NAME FOUND IN FUNCTION LIST      =0 - NOT FOUND     SEARCH     6
C**      LOC - SYMBOL TABLE LOCATION WHERE NAME WAS FOUND                 SEARCH     7
      DO 20 K=1,2                                                        SEARCH     8
      J=INITID(K)                                                        SEARCH     9
      IF(J .EQ. 0) GO TO 15                                              SEARCH    10
      DO 10 I=1,NID                                                      SEARCH    11
      IF(IDTBL(1,J) .NE. NXTID) GO TO 5                                  SEARCH    12
      ISRCH(K)=1                                                         SEARCH    13
      IDES=LOC                                                           SEARCH    14
      LOC=J                                                              SEARCH    15
      GO TO 20                                                           SEARCH    16
    5 J=IDTBL(2,J)                                                       SEARCH    17
      IF(J .EQ. 0) GO TO 15                                              SEARCH    18
   10 CONTINUE                                                           SEARCH    19
   15 ISRCH(K)=0                                                         SEARCH    20
   20 CONTINUE                                                           SEARCH    21
      RETURN                                                             SEARCH    22
      END                                                                SEARCH    23
```

FORTRAN Version

```
      SUBROUTINE SEMANT(N,FAIL)                                          SEMANT 2
      COMMON/LVARGS/LVFUNC,LVVARG,LVVAD,LVVPOS,LVVTYP,       LVVAL,
     +LVHEAD,LVVNVL,LVDEST,LVVALS(10),LVTYPE(10),LVSKIP
      COMMON/LVTABL/LVTSIZ,LVMAP(     1)/LVVSEQ/LVSIZE,LVSQSP(     1)
      COMMON/FUNC/  NARY(5,12),MARGS,IARGS(50),FNCLOC(5),NFUNC           CY58B  3
      COMMON/HL/HOL,ACTION,FUNC1,FUNC2,FUNC3,LEFT,RIGHT,STRING,MAXJ      SEMANT 4
      COMMON /TYP/ NARRAY,TYPE1,TYPE2,ERRFLG                             SEMANT 5
      COMMON /STRING/ NTYPE,NSTR,STR                                     SEMANT 6
      COMMON /JL/ JSTOP                                                  SEMANT 7
      COMMON /GIRL/NTERMS,PLUS,MINUS,SLASH,LPAR,RPAR,COMMA,STAR,EXP,LT,  SEMANT 8
     +LE,GT,GE,EQ,NE,OR,AND,NOT,EQUALS,OPRAND                            SEMANT 9
      COMMON/NEEDS/STJ,JSTACK,R,JAS,J,JLAST,RTEMP,STACK(400)             SEMANT10
      COMMON /NEED/ START,ASSOC,LEVEL,STOP                               SEMANT11
      COMMON/NOPAR/NOPAR,NDEP,NDEPTH,NFLAG                               SEMANT12
      INTEGER HOL,ACTION,FUNC,LEFT,RIGHT,STRING,RPAR,STJ,R,STACK         SEMANT13
     +,EXP,FUNC1,FUNC2,FUNC3,TYPE1,TYPE2,TYPE(5),STR(1),STOP             SEMANT14
     $ ,ALPHA,BETA,GAMMA,OPRAND,EQUALS,AND,OR,COMMA                      SEMANT15
      LOGICAL SKIP,FLAG,ERRFLG,FAIL,NOTFLG                              SEMANT16
      INTEGER FUNCRF,ZERO,BITPUT,PLUS,FL(3),BITGET                       SEMANT17
      INTEGER GETTYP,GETDIM                                              SEMANT18
      DATA FLAG/.FALSE./,FUNCRF/86/,ZERO/0/                              SEMANT19
      DATA (TYPE(I),I=1,5)/4HREAL,6HCOMPLX,6HDOUBLE,6HINTEGR,6HLOGICL/   SEMANT20
      GETTYP(II)=MOD(II,100000)/10000                                    SEMANT21
      GETDIM(II)=MOD(II,1000000)/100000                                  SEMANT22
      GO TO 25000
25001 CONTINUE
      FAIL=.FALSE.                                                       SEMANT24
      IF(N .EQ. 0) GO TO 999                                             SEMANT25
      GO TO(10,20,30,40,50,60,70,80,90,1000,1100,1200,1300,1400,1500,    SEMANT26
     $ 1600),N                                                           SEMANT27
   10 CONTINUE
      LVVTYP = 3
      LVVPOS = 1
      LVINDX = 0
      LVFUNC=          STJ
      LVVARG=            R
      CALL LVFIND(LVINDX,LVINDX,LVINDX,LVINDX)
      LV1     AAD =             R
      IF (LVVAL.NE.-1) LV1     AAD = LVVAL
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.EQ.-1) GO TO           11
               R = LV1     AAD
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.NE.-1) GO TO           12
   11 FAIL=.TRUE.                                                        SEMANT29
      RETURN                                                             SEMANT30
C PRIMARY RECOGNIZED                                                     SEMANT31
   12 IF(STJ .EQ. PLUS .OR. STJ .EQ. MINUS) GO TO 126                    SEMANT32
      IF(STJ .NE. RPAR) GO TO 121                                        SEMANT33
      JSTACK=JSTACK+1                                                    SEMANT34
      STACK(JSTACK)=SHIFT(STOP,45) .OR. SHIFT(J,15)                      SEMANT35
      NTMP=R                                                             SEMANT36
      CALL SLEVEL(SKIP)                                                  SEMANT37
      JSTACK=JSTACK-1                                                    SEMANT38
      R=NTMP                                                             SEMANT39
```

```
      JLAST=1                                                          SEMANT40
      IF(JSTOP .GT. 0) JLAST=BITGET(STACK(JSTOP),45,15)                SEMANT41
      LVVPOS =       JLAST
      LVVTYP =   3
      LVFUNC=         HOL
      LVVARG=      STRING
      CALL LVFIND(LV2       A,LV2       B,LV2       C,LV2       D)
      LV1    AAD =      STRING
      IF (LVVAL.NE.-1) LV1    AAD = LVVAL
      LV1    AAI = LV1    AAD
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=    STRING
      LVVARG=LV1    AAI
      CALL LVDLET
      LV1    AAI = LV1    AAD
      LVDEST=  0
      LV1    AAJ = TYPE1
      LVTYPE(1) =  1
      LVVALS(1) = LV1    AAJ
      LVDEST=  0
      LVVNVL =  1
      LVFUNC =     STRING
      LVVARG=LV1    AAI
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      RETURN                                                           SEMANT43
  121 CONTINUE                                                         SEMANT44
C  GET TYPE                                                            SEMANT45
      BETA=GETDIM(STR(J))                                              SEMANT46
      IF(BETA .NE. 5) GO TO 125                                        SEMANT47
C  OPERAND IS A FUNCTION REFERENCE                                     SEMANT48
      IF(NOEP .EQ. 0) GO TO 18                                         SEMANT49
      LVVPOS=-LVVPOS
      LVVTYP=  3
      LVVPOS=          1
      LVDEST=  2
      LV1    AAD = 1
      LVTYPE(1) =  1
      LVVALS(1) = LV1    AAD
      LVDEST=  2
      LVVNVL =  1
      LVFUNC =       FUNC1
      LVVARG=    OPRAND
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
   18 R=FUNCRF                                                         SEMANT51
      JSTACK=JSTACK+1                                                  SEMANT52
      STACK(JSTACK)=SHIFT(R,45) .OR. SHIFT(J+1,15)                     SEMANT53
  125 ALPHA=GETTYP(STR(J))                                             SEMANT54
      IF(TYPE1 .GE. 0) GO TO 13                                        SEMANT55
C  SET TYPE OF STATEMENT                                               SEMANT56
      TYPE1=ALPHA                                                      SEMANT57
      IF(NTYPE .EQ. 3) TYPE1=-1                                        SEMANT58
```

```
  126 CONTINUE
      LVVPOS =           J
      LVVTYP =   3
      LVFUNC=         HOL
      LVVARG=      STRING
      CALL LVFIND(LV2       E,LV2       F,LV2       G,LV2       H)
      LV1     AAI =      STRING
      IF (LVVAL.NE.-1) LV1     AAI = LVVAL
      LV1     AAK = LV1     AAI
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=     STRING
      LVVARG=LV1     AAK
      CALL LVDLET
      LV1     AAK = LV1     AAI
      LVDEST=  0
      LV1     AAL = TYPE1
      LVTYPE(1) =  1
      LVVALS(1) = LV1     AAL
      LVDEST=  0
      LVVNVL =  1
      LVFUNC =      STRING
      LVVARG=LV1     AAK
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      RETURN                                                          SEMANT60
   13 IF(FLAG) GO TO 15                                               SEMANT61
C  CHECK FOR MIXED MODE EXPRESSION                                    SEMANT62
      IF(TYPE1 .EQ. ALPHA .OR. ALPHA .EQ. 5) GO TO 16                 CY60   1
      N1=TYPE1+1                                                      SEMANT64
      N2=ALPHA+1                                                      SEMANT65
      ERRFLG=.TRUE.                                                   SEMANT66
      CALL ERROR(77,TYPE(N1),TYPE(N2))                                SEMANT67
      LVVPOS =           J
      LVVTYP =   3
      LVFUNC=         HOL
      LVVARG=      STRING
      CALL LVFIND(LV2       I,LV2       J,LV2       K,LV2       L)
      LV1     AAI =      STRING
      IF (LVVAL.NE.-1) LV1     AAI = LVVAL
      LV1     AAK = LV1     AAI
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=     STRING
      LVVARG=LV1     AAK
      CALL LVDLET
      LV1     AAK = LV1     AAI
      LVDEST=  0
      LV1     AAM = TYPE1
      LVTYPE(1) =  1
      LVVALS(1) = LV1     AAM
      LVDEST=  0
      LVVNVL =  1
      LVFUNC =      STRING
```

```
      LVVARG=LV1     AAK
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      RETURN                                                          SEMANT69
C  PARSING  AN EXPONENT                                               SEMANT70
   15 IF(ALPHA .EQ. 3 .AND. TYPE1 .EQ. 3) GO TO 16                    SEMANT71
      IF((TYPE1 .EQ. 0 .OR. TYPE1 .EQ. 2) .AND. (ALPHA .EQ. 0 .OR.    SEMANT72
     $ ALPHA .EQ. 2)) GO TO 16                                        SEMANT73
      CALL ERROR(78,J)                                                SEMANT74
      ERRFLG=.TRUE.                                                   SEMANT75
      LVVPOS =           J
      LVVTYP =   3
      LVFUNC=         HOL
      LVVARG=      STRING
      CALL LVFIND(LV2      M,LV2      N,LV2      O,LV2      P)
      LV1    AAI =      STRING
      IF (LVVAL.NE.-1) LV1    AAI = LVVAL
      LV1    AAK = LV1    AAI
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=    STRING
      LVVARG=LV1    AAK
      CALL LVDLET
      LV1    AAK = LV1    AAI
      LVDEST=  0
      LV1    AAN = TYPE1
      LVTYPE(1) =  1
      LVVALS(1) = LV1    AAN
      LVDEST=  0
      LVVNVL =  1
      LVFUNC =     STRING
      LVVARG=LV1    AAK
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      RETURN                                                          SEMANT77
   16 IF((.NOT. FLAG .AND. TYPE1 .LT. ALPHA).OR.(FLAG .AND. ALPHA .NE. 3SEMANT78
     + .AND. TYPE1 .LT. ALPHA)) TYPE1=ALPHA                           SEMANT79
      LVVPOS =           J
      LVVTYP =   3
      LVFUNC=         HOL
      LVVARG=      STRING
      CALL LVFIND(LV2      Q,LV2      R,LV2      S,LV2      T)
      LV1    AAI =      STRING
      IF (LVVAL.NE.-1) LV1    AAI = LVVAL
      LV1    AAK = LV1    AAI
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=    STRING
      LVVARG=LV1    AAK
      CALL LVDLET
      LV1    AAK = LV1    AAI
      LVDEST=  0
      LV1    AAO = TYPE1
```

```
      LVTYPE(1) =  1
      LVVALS(1) = LV1     AAO
      LVOEST=  0
      LVVNVL =  1
      LVFUNC =      STRING
      LVVARG=LV1     AAK
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      RETURN                                                        SEMANT81
C  WILL SCAN AN EXPONENT                                            SEMANT82
   20 IF(STJ .LT. 0) RETURN                                         SEMANT83
      LVVTYP = 3
      LVVPOS = 1
      LVINDX = 0
      LVFUNC=          STJ
      LVVARG=            R
      CALL LVFIND(LVINDX,LVINDX,LVINDX,LVINDX)
      LV1     AAI =            R
      IF (LVVAL.NE.-1) LV1     AAI = LVVAL
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.EQ.-1) GO TO           11
               R = LV1     AAI
      FLAG=.TRUE.                                                   SEMANT85
      RETURN                                                        SEMANT86
C  RECOGNIZED A TERM,PRODUCT OR PRIMARY PERHAPS NEEDING PARENTHESIZATIONSEMANT87
   30 CONTINUE                                                      SEMANT88
      KTMP=R                                                        SEMANT89
      IF(NDEP .EQ. 0) GO TO 34                                      SEMANT90
      LVVPOS=-LVVPOS
      LVVTYP=  3
      LVVPOS=           1
      LVOEST=  2
      LV1     AAI = 1
      LVTYPE(1) =  1
      LVVALS(1) = LV1     AAI
      LVOEST=  2
      LVVNVL =  1
      LVFUNC =       FUNC1
      LVVARG=     OPRAND
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
   34 CONTINUE                                                      SEMANT92
      ITEST=0                                                       SEMANT93
      IF(STJ .LT. 0) GO TO 31                                       SEMANT94
      LVVTYP = 3
      LVVPOS = 1
      LVINDX = 0
      LVFUNC=          STJ
      LVVARG=            R
      CALL LVFIND(LVINDX,LVINDX,LVINDX,LVINDX)
      LV1     AAK =            R
      IF (LVVAL.NE.-1) LV1     AAK = LVVAL
               R = LV1     AAK
      LVVTR = LVVAL
```

```
      LVVAL = -100
      IF (LVVTR.NE.-1) GO TO          32
   31 CONTINUE
      LVVTYP = 3
      LVVPOS = 1
      LVINDX = 0
      LVFUNC=        STOP
      LVVARG=           R
      CALL LVFIND(LVINDX,LVINDX,LVINDX,LVINDX)
      LV1    AAK =           R
      IF (LVVAL.NE.-1) LV1    AAK = LVVAL
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.EQ.-1) GO TO          39
            R = LV1    AAK
      IF(STJ .LT. 0 .AND. KTMP .EQ. 889) GO TO 38                          SEMANT97
      ITEST=-1                                                             SEMANT98
   32 CONTINUE                                                             SEMANT99
C IF UNARY PLUS OR MINUS RETURN                                            SEMAN100
      ISTCK=BITGET(STACK(JSTOP),15,15)                                     SEMAN101
      IF(ISTCK .NE. 288 .AND. ISTCK .NE. 110) GO TO 33                     SEMAN102
      JLAST=BITGET(STACK(JSTOP),45,15)-1                                   SEMAN103
      IF(ISTCK .EQ. 288) JLAST=JLAST-1                                     SEMAN104
      LVVPOS =       JLAST
      LVVTYP =    3
      LVFUNC=         HOL
      LVVARG=      STRING
      CALL LVFIND(LV2       U,LV2       V,LV2       W,LV2       X)
      LV1    AAK =      STRING
      IF (LVVAL.NE.-1) LV1    AAK = LVVAL
      LV1    AAP = LV1    AAK
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=     STRING
      LVVARG=LV1    AAP
      CALL LVDLET
      LV1    AAP = LV1    AAK
      LVDEST=  0
      LV1    AAQ = TYPE1
      LVTYPE(1) =  1
      LVVALS(1) = LV1    AAQ
      LVDEST=  0
      LVVNVL =  1
      LVFUNC =      STRING
      LVVARG=LV1    AAP
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      LVVPOS =           J
      LVVTYP =    3
      LVFUNC=         HOL
      LVVARG=      STRING
      CALL LVFIND(LV2       Y,LV2       Z,LV2       0,LV2       1)
      LV1    AAK =      STRING
      IF (LVVAL.NE.-1) LV1    AAK = LVVAL
      LV1    AAP = LV1    AAK
```

```
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=     STRING
      LVVARG=LV1     AAP
      CALL LVDLET
      LV1     AAP = LV1     AAK
      LVDEST=  0
      LV1     AAR = TYPE1
      LVTYPE(1) =  1
      LVVALS(1) = LV1     AAR
      LVDEST=  0
      LVVNVL =  1
      LVFUNC =      STRING
      LVVARG=LV1     AAP
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      IF(ITEST .LT. 0) J=J-1                                          SEMAN107
      RETURN                                                          SEMAN108
   33 CONTINUE                                                        SEMAN109
      IF(ITEST .LT. 0) J=J-1                                          SEMAN110
      JSTACK=JSTACK+1                                                 SEMAN111
      STACK(JSTACK)=SHIFT(STOP,45) .OR. SHIFT(J,15)                   SEMAN112
      NTMP=R                                                          SEMAN113
      CALL SLEVEL(SKIP)                                               SEMAN114
      JSTACK=JSTACK-1                                                 SEMAN115
      R=NTMP                                                          SEMAN116
      JLAST=1                                                         SEMAN117
      IF(JSTOP .GT. 0) JLAST=BITGET(STACK(JSTOP),45,15)               SEMAN118
      NTMP=TYPE1                                                      SEMAN119
      JJ=JLAST                                                        SEMAN120
      IF(BITGET(STACK(JSTACK),15,15) .EQ. 418 .AND. JLAST .GT. 1)     SEMAN121
     $ JJ=JLAST-1                                                     SEMAN122
      LVVPOS =          JJ
      LVVTYP =   3
      LVFUNC=         HOL
      LVVARG=     STRING
      CALL LVFIND(LV2      2,LV2      3,LV2      4,LV2      5)
      LV1     AAK =      STRING
      IF (LVVAL.NE.-1) LV1     AAK = LVVAL
      LVVTYP = 3
      LVVPOS = 1
      LVINDX = 0
      LVFUNC=     STRING
      LVVARG= LV1     AAK
      CALL LVFIND(LVINDX,LVINDX,LVINDX,LVINDX)
      LV1     AAP = LV1     AAK
      IF (LVVAL.NE.-1) LV1     AAP = LVVAL
           TYPE1 = LV1     AAP
      IF(.NOT. FLAG .OR. NTMP .EQ. 3) GO TO 35                        SEMAN124
      IF((NTMP.EQ.0.OR.NTMP.EQ.2).AND.(TYPE1.EQ.0.OR.TYPE1.EQ.2))GOTO 35SEMAN125
      ERRFLG=.TRUE.                                                   SEMAN126
      CALL ERROR(78,J)                                                SEMAN127
   35 CONTINUE                                                        SEMAN128
      IF(TYPE1 .GT. 2 .OR. NTYPE .GT. 1) GO TO 38                     SEMAN129
      FUNC=FUNC1                                                      SEMAN130
```

```
      IF(TYPE1 .EQ. 1) FUNC=FUNC2                                          SEMAN131
      IF(TYPE1 .EQ. 2) FUNC=FUNC3                                          SEMAN132
      LVVPOS =       JLAST
      LVVTYP =    3
      LVFUNC=         HOL
      LVVARG=      STRING
      CALL LVFIND(LV2       6,LV2       7,LV2       8,LV2       9)
      LV1     AAP =      STRING
      IF (LVVAL.NE.-1) LV1     AAP = LVVAL
      LV1     AAK = LV1     AAP
      LV1     AAS =        LEFT
      LVVTYP=  3
      LVVPOS=           1
      LVDEST=  1
      LVTYPE(1) =  0
      LVVALS(1) =        LPAR
      LVVNVL =  1
      LVFUNC = LV1     AAS
      LVVARG=LV1     AAK
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      LV1     AAK = LV1     AAP
      LVVTYP=  3
      LVVPOS=           1
      LVDEST=  1
      LVTYPE(1) =  0
      LVVALS(1) =        FUNC
      LVVNVL =  1
      LVFUNC = LV1     AAS
      LVVARG=LV1     AAK
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      LVVPOS =           J
      LVVTYP =    3
      LVFUNC=         HOL
      LVVARG=      STRING
      CALL LVFIND(LV2      AA,LV2      AB,LV2      AC,LV2      AD)
      LV1     AAP =      STRING
      IF (LVVAL.NE.-1) LV1     AAP = LVVAL
      LVDEST=  0
      LVTYPE(1) =  0
      LVVALS(1) =        RPAR
      LVVNVL =  1
      LVFUNC =       RIGHT
      LVVARG=LV1     AAP
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
   38 IF(ITEST .LT. 0 .AND. STJ .LT. 0) J=J+1                              SEMAN135
      FLAG=.FALSE.                                                         SEMAN136
      RETURN                                                               SEMAN137
   39 FLAG=.FALSE.                                                         SEMAN138
      GO TO 11                                                             SEMAN139
C  CHECK FOR CORRECTNESS OF SUBSCRIPTS AND DO IMPLIED LIST PARAMETERS      SEMAN140
   40 NR=R                                                                 SEMAN141
```

```
      LVVTYP = 3
      LVVPOS = 1
      LVINDX = 0
      LVFUNC=          STJ
      LVVARG=            R
      CALL LVFIND(LVINDX,LVINDX,LVINDX,LVINDX)
      LV1     AAP =             R
      IF (LVVAL.NE.-1) LV1     AAP = LVVAL
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.EQ.-1) GO TO           11
              R = LV1     AAP
      NBETA=GETDIM(STR(J))                                                    SEMAN143
      IF(NR .EQ. 359 .AND. NBETA .NE. 4) GO TO 47                             SEMAN144
      ALPHA=GETTYP(STR(J))                                                    SEMAN145
      GAMMA=STR(J)/1000000                                                    SEMAN146
      IF(NTYPE .EQ. 3) GO TO 45                                               SEMAN147
      IF(NR .EQ. 839 .AND. NBETA .EQ. 0) GO TO 45                             SEMAN148
      IF(NR .EQ. 359) GO TO 45                                                SEMAN149
      IF(NR .EQ. 21 .AND. NBETA .EQ. 4) GO TO 45                              SEMAN150
      IF(NBETA .EQ. 0 .AND. NR .EQ. 935) GO TO 45                             SEMAN151
      IF(NTYPE .EQ. 2 .AND. NR .EQ. 935 .AND. STR(J-1) .NE. -7) GO TO 45SEMAN152
      IF(NTYPE .EQ. 2 .AND. NR .EQ. 359 .AND. NBETA .EQ. 0) GO TO 11          SEMAN153
      CALL ERROR(79,J)                                                        SEMAN154
      ERRFLG=.TRUE.                                                           SEMAN155
45    CONTINUE                                                                SEMAN156
      IF(GAMMA .GE. 6 .AND. NBETA .EQ. 4) CALL ERROR(76)                      SEMAN157
      IF(ALPHA .EQ. 3) GO TO 46                                               SEMAN158
      N1=ALPHA+1                                                              SEMAN159
      ERRFLG=.TRUE.                                                           SEMAN160
      CALL ERROR(80,TYPE(N1),J)                                               SEMAN161
46    IF(NBETA .EQ. 4) RETURN                                                 SEMAN162
      MARGS=MARGS+1                                                           SEMAN163
      LV1     AAP =       OPRAND
      LVDEST=  0
      LV1     AAS = MARGS
      LVTYPE(1) =  1
      LVVALS(1) = LV1     AAS
      LVDEST=  0
      LVVNVL =  1
      LVFUNC =       FUNC2
      LVVARG=LV1     AAP
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      LVDEST=  0
      LV1     AAK = J
      LVTYPE(1) =  1
      LVVALS(1) = LV1     AAK
      LVDEST=  0
      LVVNVL =  1
      LVFUNC =       FUNC3
      LVVARG=LV1     AAP
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      LVDEST=  0
```

```
      LV1    AAT = NDEPTH
      LVTYPE(1) =  1
      LVVALS(1) = LV1    AAT
      LVDEST=  0
      LVVNVL =  1
      LVFUNC =       LEVEL
      LVVARG=LV1     AAP
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      IVR=(MARGS+2)/3                                              SEMAN165
      IF(IVR .GT. 50) GO TO 1610                                   CY58B  5
      ICOL=20*MOD(MARGS-1,3)+10                                    SEMAN166
      IVAL=MOD(STR(J),10000)                                       SEMAN167
      IARGS(IVR)=BITPUT(IARGS(IVR),IVAL,ICOL)                      SEMAN168
      IF(NR .EQ. 839) GO TO 49                                     SEMAN169
      IF(NTYPE .NE. 3) RETURN                                      SEMAN170
C  FLAG SUBSCRIPT IN I/O LIST                                      SEMAN171
      IARGS(IVR)=BITPUT(IARGS(IVR),1,ICOL+5)                       SEMAN172
      RETURN                                                       SEMAN173
C FLAG DO INDEX IN I/O LIST                                        SEMAN174
   49 IARGS(IVR)=BITPUT(IARGS(IVR),2,ICOL+5)                       SEMAN175
      IF(NFLAG .LT. 1) RETURN                                      SEMAN176
      IARGS(IVR)=BITPUT(IARGS(IVR),FL(NFLAG),ICOL+10)              SEMAN177
      RETURN                                                       SEMAN178
   47 NR=R                                                         SEMAN179
C    SUBSCRIPT DOES NOT BEGIN WITH CONSTANT, FORCE SEARCH FOR VARIABLE SEMAN180
      GO TO 11                                                     SEMAN181
C  CHECK FOR PROPER NUMBER OF SUBSCRIPTS                           SEMAN182
   50 IF( BETA .EQ. 4 .OR. R .NE. 452) GO TO 52                    SEMAN183
      MARGS=MARGS+1                                                SEMAN184
      LV1    AAP =     OPRAND
      LVDEST=  0
      LV1    AAU = MARGS
      LVTYPE(1) =  1
      LVVALS(1) = LV1    AAU
      LVDEST=  0
      LVVNVL =  1
      LVFUNC =       FUNC2
      LVVARG=LV1     AAP
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      LVDEST=  0
      LV1    AAV = J-1
      LVTYPE(1) =  1
      LVVALS(1) = LV1    AAV
      LVDEST=  0
      LVVNVL =  1
      LVFUNC =       FUNC3
      LVVARG=LV1     AAP
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      LVDEST=  0
      LV1    AAW = NDEPTH
      LVTYPE(1) =  1
```

```
      LVVALS(1) = LV1     AAW
      LVDEST=  0
      LVVNVL =  1
      LVFUNC =       LEVEL
      LVVARG=LV1     AAP
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      IVR=(MARGS+2)/3                                                  SEMAN186
      IF(IVR .GT. 50) GO TO 1610                                       CY588  6
      ICOL=20*MOD(MARGS-1,3)+10                                        SEMAN187
      IVAL=MOD(STR(J-1),10000)                                         SEMAN188
      IARGS(IVR)=BITPUT(IARGS(IVR),IVAL,ICOL)                          SEMAN189
      IF(NOPAR .LE. 0) GO TO 52                                        SEMAN190
      LVVPOS =           1
      LVVTYP =   3
      LVVPOS=-LVVPOS
      LVFUNC=      ACTION
      LVVARG=      OPRAND
      CALL LVFIND(LV2     AE,LV2     AF,LV2     AG,LV2     AH)
      LV1    AAP =      OPRAND
      IF (LVVAL.NE.-1) LV1    AAP = LVVAL
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.EQ.-1) GO TO          52
           MFUNC = LV1    AAP
      IARGS(IVR)=BITPUT(IARGS(IVR),MFUNC ,ICOL+3)                      SEMAN192
      IARGS(IVR)=BITPUT(IARGS(IVR),NARGS,ICOL+9)                       SEMAN193
C IF NO STRING LEFT, RETURN IF CONSTANT,VARIABLE OR I/O LIST           SEMAN194
   52 IF(STJ .LT. 0 .AND. (BETA .EQ. 0 .OR. BETA .EQ. 4               SEMAN195
     $ .OR. NTYPE .EQ. 3)) RETURN                                      SEMAN196
      IF(BETA .GT. NARRAY) GO TO 55                                    SEMAN197
      ERRFLG=.TRUE.                                                    SEMAN198
      CALL ERROR(81,J)                                                 SEMAN199
   55 IF(R .EQ. 452) NARRAY=0                                          SEMAN200
      IF(R .EQ. 318) NARRAY=1                                          SEMAN201
      IF(R .EQ.  60) NARRAY=2                                          SEMAN202
      IF(R .EQ. 103) NARRAY=3                                          SEMAN203
      IF(STJ .LT. 0) GO TO 58                                          SEMAN204
      LVVTYP = 3
      LVVPOS = 1
      LVINDX = 0
      LVFUNC=         STJ
      LVVARG=           R
      CALL LVFIND(LVINDX,LVINDX,LVINDX,LVINDX)
      LV1    AAP =           R
      IF (LVVAL.NE.-1) LV1    AAP = LVVAL
           R = LV1    AAP
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.NE.-1) GO TO          56
   58 IF(NTYPE .EQ. 3 .AND. NARRAY .EQ. 0) GO TO 57                    SEMAN206
      IF(BETA .GE. 1 .AND. BETA .LE. 3 .AND. NOPAR .EQ. 0)             SEMAN207
     * CALL ERROR(82,J)                                               SEMAN208
   57 NARRAY=-1                                                        SEMAN209
      GO TO 11                                                         SEMAN210
   56 IF(NTYPE .EQ. 3 .AND. NARRAY .EQ. 0) RETURN                      SEMAN211
```

```
      IF(STJ .EQ. RPAR .AND. NARRAY .LT.  BETA .AND. J .EQ. MAXJ)        SEMAN212
     $ CALL ERROR(82,J)                                                  SEMAN213
      IF(STJ .EQ. RPAR) NARRAY=-1                                        SEMAN214
      RETURN                                                             SEMAN215
C  RESET TYPE OF STATEMENT IN ANTICIPATION OF SEARCH FOR BOOLEAN PRIMARY SEMAN216
   60 CONTINUE                                                           SEMAN217
      NOTFLG=.FALSE.                                                     SEMAN218
      IF(STR(J-1) .EQ. -17) NOTFLG=.TRUE.                                SEMAN219
      TYPE1=-1                                                           SEMAN220
      LVVPOS =           J
      LVVTYP =   3
      LVFUNC=         HOL
      LVVARG=     STRING
      CALL LVFIND(LV2     AI,LV2     AJ,LV2     AK,LV2     AL)
      LV1    AAP =     STRING
      IF (LVVAL.NE.-1) LV1    AAP = LVVAL
      LV1    AAX = LV1    AAP
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=    STRING
      LVVARG=LV1    AAX
      CALL LVDLET
      LV1    AAX = LV1    AAP
      LVDEST=  0
      LV1    AAY = TYPE1
      LVTYPE(1) =  1
      LVVALS(1) = LV1    AAY
      LVDEST=  0
      LVVNVL =  1
      LVFUNC =     STRING
      LVVARG=LV1    AAX
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      IF(STJ .NE. OPRAND) GO TO 65                                       SEMAN222
      ALPHA=GETTYP(STR(J))                                               SEMAN223
      BETA=GETDIM(STR(J))                                                SEMAN224
      IF(ALPHA .NE. 4) GO TO 11                                          SEMAN225
   65 CONTINUE                                                           SEMAN226
      LVVTYP = 3
      LVVPOS = 1
      LVINDX = 0
      LVFUNC=         STJ
      LVVARG=           R
      CALL LVFIND(LVINDX,LVINDX,LVINDX,LVINDX)
      LV1    AAP =          R
      IF (LVVAL.NE.-1) LV1    AAP = LVVAL
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.EQ.-1) GO TO          11
            R = LV1    AAP
      RETURN                                                             SEMAN228
C  IF BOOLEAN PRIMARY IS AN ARETHMETIC COMPARE CONTINUE PARSING PRIMARY  SEMAN229
   70 IF(STJ .LT. 0) RETURN                                              SEMAN230
      IF(TYPE1 .EQ. 4) GO TO 75                                          SEMAN231
      LVVTYP = 3
```

```
      LVVPOS = 1
      LVINDX = 0
      LVFUNC=         STJ
      LVVARG=           R
      CALL LVFIND(LVINDX,LVINDX,LVINDX,LVINDX)
      LV1    AAP =           R
      IF (LVVAL.NE.-1) LV1    AAP = LVVAL
                R = LV1    AAP
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.EQ.-1) GO TO          11
C  RELATIONAL OPERATOR FOUND                                          SEMAN233
      IF(NDEP .EQ. 0) RETURN                                         SEMAN234
      LVVPOS=-LVVPOS
      LVVTYP=  3
      LVVPOS=          1
      LVDEST=  2
      LV1    AAP = 1
      LVTYPE(1) =  1
      LVVALS(1) = LV1    AAP
      LVDEST=  2
      LVVNVL =  1
      LVFUNC =       FUNC1
      LVVARG=    OPRAND
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      RETURN                                                         SEMAN236
C  IF BOOLEAN VARIABLE OR CONSTANT, SET STATE TO STOP                 SEMAN237
   75 P=STOP                                                         SEMAN238
      JSTACK=JSTACK+1                                                SEMAN239
      STACK(JSTACK)=SHIFT(P,45) .OR. SHIFT(J,15)                     SEMAN240
      GO TO 11                                                       SEMAN241
C  COMPARE TYPES ON BOTH SIDES OF RELATIONAL EXPRESSION               SEMAN242
   80 IF(TYPE1 .EQ. 0 .OR. TYPE1 .EQ. 2 .OR. TYPE1 .EQ. 3) GO TO 85  SEMAN243
      ERRFLG=.TRUE.                                                  SEMAN244
      CALL ERROR(83,J)                                               SEMAN245
      TYPE1=-1                                                       SEMAN246
      LVVPOS =          J
      LVVTYP =   3
      LVFUNC=         HOL
      LVVARG=     STRING
      CALL LVFIND(LV2     AM,LV2     AN,LV2     AO,LV2     AP)
      LV1    AAX =     STRING
      IF (LVVAL.NE.-1) LV1    AAX = LVVAL
      LV1    AAZ = LV1    AAX
      LVVAO=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=    STRING
      LVVARG=LV1    AAZ
      CALL LVDLET
      LV1    AAZ = LV1    AAX
      LVDEST=  0
      LV1    AA0 = TYPE1
      LVTYPE(1) =  1
      LVVALS(1) = LV1    AA0
```

```
      LVDEST=  0                                                          SEMAN247
      LVVNVL =  1                                                         SEMAN248
      LVFUNC =     STRING                                                 SEMAN249
      LVVARG=LV1    AAZ                                                   SEMAN250
      CALL LVNSRT                                                         SEMAN251
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)                                   SEMAN252
      IF(LVVAL.LT.0) RETURN                                               SEMAN253
   85 TYPE2=TYPE1                                                         SEMAN254
      GO TO 11                                                            SEMAN255
C  BOOLEAN PRIMARY RECOGNIZED-SET TYPE TO BOOLEAN AND CONTINUE PARSE      SEMAN256
   90 IF(TYPE1 .EQ. TYPE2 .OR. TYPE1+TYPE2 .EQ. 2 .OR.                    SEMAN257
     + TYPE2 .LT. 0) GO TO 95                                             SEMAN258
      N1=TYPE1+1                                                          SEMAN259
      N2=TYPE2+1                                                          SEMAN260
      CALL ERROR(77,TYPE(N1),TYPE(N2))                                    SEMAN261
      ERRFLG=.TRUE.                                                       SEMAN262
   95 TYPE1=4                                                             SEMAN263
      TYPE2=-1                                                            SEMAN264
      IF(STJ .LT. 0) RETURN                                               SEMAN265
      LVVPOS =           J
      LVVTYP =   3
      LVFUNC=         HOL
      LVVARG=     STRING
      CALL LVFIND(LV2     AQ,LV2     AR,LV2     AS,LV2     AT)
      LV1    AAX =     STRING
      IF (LVVAL.NE.-1) LV1    AAX = LVVAL
      LV1    AAZ = LV1    AAX
      LVVAO=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=    STRING
      LVVARG=LV1    AAZ
      CALL LVDLET
      LV1    AAZ = LV1    AAX
      LVDEST=  0
      LV1    AA1 = TYPE1
      LVTYPE(1) =  1
      LVVALS(1) = LV1    AA1
      LVDEST=  0
      LVVNVL =  1
      LVFUNC =     STRING
      LVVARG=LV1    AAZ
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      GO TO 11                                                            SEMAN266
C  PARSE REACHED BLIND ALLEY-MUST BACK UP AND REMOVE PARENTHESES CREATED  SEMAN267
  999 JM=BITGET(STACK(JSTACK),45,15)                                      SEMAN268
      K=JM                                                                SEMAN269
      DO 996 KK=JM,J                                                      SEMAN270
      LVVPOS =           K
      LVVTYP =   3
      LVFUNC=         HOL
      LVVARG=     STRING
      CALL LVFIND(LV2     AU,LV2     AV,LV2     AW,LV2     AX)
      LV1    AAX =     STRING
      IF (LVVAL.NE.-1) LV1    AAX = LVVAL
```

```
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.EQ.-1) GO TO         995
      LVVPOS =            1
      LVVTYP =    3
      LVFUNC=      STRING
      LVVARG= LV1     AAX
      CALL LVFIND(LV2      AY,LV2      AZ,LV2      A0,LV2      A1)
      LV1     AAZ = LV1     AAX
      IF (LVVAL.NE.-1) LV1     AAZ = LVVAL
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.EQ.-1) GO TO         996
         TYPE1 = LV1     AAZ
      GO TO 995                                                    SEMAN267
  996 K=K-1                                                        SEMAN268
  995 CONTINUE                                                     SEMAN269
      DO 998 I=JM,J                                                SEMAN270
      LVVPOS =            I
      LVVTYP =    3
      LVFUNC=          HOL
      LVVARG=      STRING
      CALL LVFIND(LV2      A2,LV2      A3,LV2      A4,LV2      A5)
      LV1     AAZ =       STRING
      IF (LVVAL.NE.-1) LV1     AAZ = LVVAL
      LV1     AAX = LV1     AAZ
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=       LEFT
      LVVARG=LV1     AAX
      CALL LVDLET
      LV1     AAX = LV1     AAZ
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=      RIGHT
      LVVARG=LV1     AAX
      CALL LVDLET
  998 CONTINUE                                                     SEMAN272
  980 CONTINUE
      LVVPOS =            1
      LVVTYP =    3
      LVVPOS=-LVVPOS
      LVFUNC=       FUNC3
      LVVARG=      OPRAND
      CALL LVFIND(LV2      A6,LV2      A7,LV2      A8,LV2      A9)
      LV1     AAZ =      OPRAND
      IF (LVVAL.NE.-1) LV1     AAZ = LVVAL
            JN = LV1     AAZ
      IF(JN .LT. JM) GO TO 985                                     SEMAN274
      LV1     AAZ =      OPRAND
      LVVPOS =            1
      LVVTYP =    3
      LVVPOS=-LVVPOS
      LVFUNC=       FUNC2
      LVVARG= LV1     AAZ
```

```
      LVVAD=-1
      CALL LVDLET
      LVVPOS =           1
      LVVTYP =   3
      LVVPOS=-LVVPOS
      LVFUNC=       FUNC3
      LVVARG= LV1     AAZ
      LVVAD=-1
      CALL LVDLET
      LVVPOS =           1
      LVVTYP =   3
      LVVPOS=-LVVPOS
      LVFUNC=       LEVEL
      LVVARG= LV1     AAZ
      LVVAD=-1
      CALL LVDLET
      GO TO 980                                                    SEMAN276
  985 CONTINUE
      LVVPOS =           1
      LVVTYP =   3
      LVVPOS=-LVVPOS
      LVFUNC=       FUNC2
      LVVARG=     OPRAND
      CALL LVFIND(LV2     BA,LV2     BB,LV2     BC,LV2     BD)
      LV1     AAZ =     OPRAND
      IF (LVVAL.NE.-1) LV1     AAZ = LVVAL
           MARGS = LV1     AAZ
      LVVPOS =           1
      LVVTYP =   3
      LVVPOS=-LVVPOS
      LVFUNC=       LEVEL
      LVVARG=     OPRAND
      CALL LVFIND(LV2     BE,LV2     BF,LV2     BG,LV2     BH)
      LV1     AAZ =     OPRAND
      IF (LVVAL.NE.-1) LV1     AAZ = LVVAL
          NDEPTH = LV1     AAZ
      RETURN                                                       SEMAN279
                                                                   SEMAN280
C   RECOGNIZED FUNCTION-PREPARE TO SET TYPE OF ARGUMENTS FOR THE 'NDEPTHSEMAN281
C           FUNCTION IN THIS STMT                                   SEMAN282
 1000 CONTINUE                                                     SEMAN283
      NDEPTH=NDEPTH+1                                              SEMAN284
      NDEP=NDEP+1                                                  SEMAN285
      NARGS=0                                                      SEMAN286
      LVVTYP = 3
      LVVPOS = 1
      LVINDX = 0
      LVFUNC=         STJ
      LVVARG=           R
      CALL LVFIND(LVINDX,LVINDX,LVINDX,LVINDX)
      LV1     AAZ =           R
      IF (LVVAL.NE.-1) LV1     AAZ = LVVAL
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.EQ.-1) GO TO          11
            R = LV1     AAZ
      NARGS=1                                                      SEMAN288
```

```
      LV1    AAZ =      OPRAND
      LVDEST=  0
      LV1    AAX = TYPE1
      LVTYPE(1) =  1
      LVVALS(1) = LV1    AAX
      LVDEST=  0
      LVVNVL =  1
      LVFUNC =      OPRAND
      LVVARG=LV1     AAZ
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      LVDEST=  0
      LV1    AA2 = NARGS
      LVTYPE(1) =  1
      LVVALS(1) = LV1    AA2
      LVDEST=  0
      LVVNVL =  1
      LVFUNC =      STRING
      LVVARG=LV1     AAZ
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      LVDEST=  0
      LV1    AA3 = NDEPTH
      LVTYPE(1) =  1
      LVVALS(1) = LV1    AA3
      LVDEST=  0
      LVVNVL =  1
      LVFUNC =      ACTION
      LVVARG=LV1     AAZ
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      LVDEST=  0
      LV1    AAZ = 0
      LVTYPE(1) =  1
      LVVALS(1) = LV1    AAZ
      LVDEST=  0
      LVVNVL =  1
      LVFUNC =      FUNC1
      LVVARG=    OPRAND
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      TYPE1=-1                                                    SEMAN291
      NOPAR=NOPAR+1                                               SEMAN292
      RETURN                                                      SEMAN293
                                                                  SEMAN294
C KEEP TRACK OF THE NUMBER AND TYPES OF ARGUMENTS IN FUNCTION CALLS SEMAN295
C    MUST USE STACK FOR POSSIBLE RECURSIVE FUNCTION USE          SEMAN296
 1100 CONTINUE                                                    SEMAN297
      LVVTYP = 3
      LVVPOS = 1
      LVINDX = 0
      LVFUNC=         STJ
      LVVARG=           R
```

```
      CALL LVFIND(LVINDX,LVINDX,LVINDX,LVINDX)
      LV1    AA4 =           R
      IF (LVVAL.NE.-1) LV1    AA4 = LVVAL
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.EQ.-1) GO TO          11
               R = LV1    AA4
      LV1    AA4 =     OPRAND
      LVVPOS =           1
      LVVTYP =   3
      LVVPOS=-LVVPOS
      LVFUNC=     STRING
      LVVARG= LV1    AA4
      CALL LVFIND(LV2     BI,LV2     BJ,LV2     BK,LV2     BL)
      LV1    AA5 = LV1    AA4
      IF (LVVAL.NE.-1) LV1    AA5 = LVVAL
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.EQ.-1) GO TO        1103
           NARGS = LV1    AA5
      LVVPOS =           1
      LVVTYP =   3
      LVVPOS=-LVVPOS
      LVFUNC=     STRING
      LVVARG= LV1    AA4
      LVVAD=-1
      CALL LVDLET
      LVVPOS =           1
      LVVTYP =   3
      LVVPOS=-LVVPOS
      LVFUNC=     ACTION
      LVVARG= LV1    AA4
      CALL LVFIND(LV2     BM,LV2     BN,LV2     BO,LV2     BP)
      LV1    AA5 = LV1    AA4
      IF (LVVAL.NE.-1) LV1    AA5 = LVVAL
           MFUNC = LV1    AA5
      LVVPOS =           1
      LVVTYP =   3
      LVVPOS=-LVVPOS
      LVFUNC=      FUNC1
      LVVARG=     OPRAND
      CALL LVFIND(LV2     BQ,LV2     BR,LV2     BS,LV2     BT)
      LV1    AA4 =     OPRAND
      IF (LVVAL.NE.-1) LV1    AA4 = LVVAL
           IEXP = LV1    AA4
 1103 CONTINUE                                                    SEMAN301
C  STORE ARGUMENT TYPES                                          SEMAN302
      IF(NDEPTH .GT. 5) CALL ERROR(85)                           SEMAN303
      IF(NDEPTH .GT. 5) GO TO 1130                               SEMAN304
      IF(NARGS .LE. 63) GO TO 1104                               SEMAN305
      ERRFLG=.TRUE.                                              SEMAN306
      CALL ERROR(84,NDEPTH)                                      SEMAN307
      GO TO 11                                                   SEMAN308
 1104 CONTINUE                                                   SEMAN309
      MM=(11+NARGS)/6                                            SEMAN310
      ITEMP=NARGS-6*(MM-2)                                       SEMAN311
      ICOL=9*ITEMP-6                                             SEMAN312
```

```
      NARY(MFUNC,MM)=BITPUT(NARY(MFUNC,MM),MOD((TYPE1+1),6),ICOL)        CY60   2
      IF(STR(J-2) .NE. -6 .AND. STR(J-2) .NE. -4) GO TO 1130             SEMAN314
      NDIM=GETDIM(STR(J-1))                                              SEMAN315
      IF(NDIM .GE. 4) GO TO 1130                                         SEMAN316
C  STORE DIMENSIONALITY OF ARGUMENTS                                     SEMAN317
      NARY(MFUNC ,MM)=BITPUT(NARY(MFUNC ,MM),NDIM,ICOL+3)                SEMAN318
 1131 CONTINUE                                                           SEMAN319
      NARY(MFUNC ,MM)=BITPUT(NARY(MFUNC ,MM),IEXP,54+ITEMP)              SEMAN320
      IF(STJ .EQ. COMMA) GO TO 1105                                      SEMAN321
      NARY(MFUNC,1)=NARGS                                                SEMAN322
      LV1    AA4 =      OPRAND
      LVVPOS =           1
      LVVTYP =   3
      LVVPOS=-LVVPOS
      LVFUNC=     ACTION
      LVVARG= LV1    AA4
      LVVAD=-1
      CALL LVDLET
      LVVPOS =           1
      LVVTYP =   3
      LVVPOS=-LVVPOS
      LVFUNC=     OPRAND
      LVVARG= LV1    AA4
      CALL LVFIND(LV2      BU,LV2      BV,LV2      BW,LV2      BX)
      LV1    AA5 = LV1    AA4
      IF (LVVAL.NE.-1) LV1    AA5 = LVVAL
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.EQ.-1) GO TO        1135
           TYPE1 = LV1    AA5
      LV1    AA4 =      OPRAND
      LVVPOS =           1
      LVVTYP =   3
      LVVPOS=-LVVPOS
      LVFUNC=       FUNC1
      LVVARG= LV1    AA4
      LVVAD=-1
      CALL LVDLET
      LVVPOS =           1
      LVVTYP =   3
      LVVPOS=-LVVPOS
      LVFUNC=     OPRAND
      LVVARG= LV1    AA4
      LVVAD=-1
      CALL LVDLET
 1135 NOPAR=NOPAR-1                                                      SEMAN325
      NDEP=NDEP-1                                                        SEMAN326
      RETURN                                                             SEMAN327
 1105 TYPE1=-1                                                           SEMAN328
      LVVPOS =           J
      LVVTYP =   3
      LVFUNC=         HOL
      LVVARG=     STRING
      CALL LVFIND(LV2      BY,LV2      BZ,LV2      B0,LV2      B1)
      LV1    AA4 =      STRING
      IF (LVVAL.NE.-1) LV1    AA4 = LVVAL
      LV1    AA5 = LV1    AA4
```

```
      LVVAD=-1
      LVVTYP=-1
      LVVPOS=1
      LVFUNC=     STRING
      LVVARG=LV1     AA5
      CALL LVDLET
      LV1     AA5 = LV1     AA4
      LVDEST=  0
      LV1     AA6 = TYPE1
      LVTYPE(1) =  1
      LVVALS(1) = LV1     AA6
      LVDEST=  0
      LVVNVL =  1
      LVFUNC =      STRING
      LVVARG=LV1     AA5
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      NARGS=NARGS+1                                                    SEMAN330
      LVDEST=  0
      LV1     AA4 = NARGS
      LVTYPE(1) =  1
      LVVALS(1) = LV1     AA4
      LVDEST=  0
      LVVNVL =  1
      LVFUNC =      STRING
      LVVARG=    OPRAND
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      LVVPOS=-LVVPOS
      LVVTYP=  3
      LVVPOS=          1
      LVDEST=  2
      LV1     AA5 = 0
      LVTYPE(1) =  1
      LVVALS(1) = LV1     AA5
      LVDEST=  2
      LVVNVL =  1
      LVFUNC =       FUNC1
      LVVARG=    OPRAND
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      RETURN                                                           SEMAN333
C  SAVE TYPE OF STAEMENT WHILE PARSING EXPONENT                        SEMAN334
 1200 CONTINUE                                                         SEMAN335
      LVVPOS =           J
      LVVTYP =   3
      LVFUNC=         HOL
      LVVARG=     STRING
      CALL LVFIND(LV2     B2,LV2     B3,LV2     B4,LV2     B5)
      LV1     AA7 =      STRING
      IF (LVVAL.NE.-1) LV1     AA7 = LVVAL
      LV1     AA8 = LV1     AA7
      LVVAD=-1
      LVVTYP=-1
```

```
      LVVPOS=1
      LVFUNC=     STRING
      LVVARG=LV1     AA8
      CALL LVDLET
      LV1     AA8 = LV1     AA7
      LVDEST=  0
      LV1     AA9 = TYPE1
      LVTYPE(1) =  1
      LVVALS(1) = LV1     AA9
      LVDEST=  0
      LVVNVL =  1
      LVFUNC =      STRING
      LVVARG=LV1     AA8
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      TYPE1=-1                                                          SEMAN337
      LVVTYP = 3
      LVVPOS = 1
      LVINDX = 0
      LVFUNC=          STJ
      LVVARG=            R
      CALL LVFIND(LVINDX,LVINDX,LVINDX,LVINDX)
      LV1     AA7 =             R
      IF (LVVAL.NE.-1) LV1     AA7 = LVVAL
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.EQ.-1) GO TO          11
            R = LV1     AA7
      RETURN                                                            SEMAN339
 1300 CONTINUE                                                          SEMAN340
      IF(STJ .LT. 0) RETURN                                             SEMAN341
      IF(STJ .NE. AND .AND. STJ .NE. OR .AND. STJ .NE. NOT) GO TO 11    SEMAN342
      LVVTYP = 3
      LVVPOS = 1
      LVINDX = 0
      LVFUNC=          STJ
      LVVARG=            R
      CALL LVFIND(LVINDX,LVINDX,LVINDX,LVINDX)
      LV1     AA7 =             R
      IF (LVVAL.NE.-1) LV1     AA7 = LVVAL
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.EQ.-1) GO TO          11
            R = LV1     AA7
      IF(NDEP .EQ. 0) RETURN                                            SEMAN344
C  LOGICAL OPERATOR FOUND                                               SEMAN345
      LVVPOS=-LVVPOS
      LVVTYP=  3
      LVVPOS=          1
      LVDEST=  2
      LV1     AA7 = 1
      LVTYPE(1) =  1
      LVVALS(1) = LV1     AA7
      LVDEST=  2
      LVVNVL =  1
      LVFUNC =       FUNC1
```

```
      LVVARG=     OPRAND
      CALL LVNSRT
      IF(LVVAL.LT.0) CALL LVEXIT(LVVAL)
      IF(LVVAL.LT.0) RETURN
      RETURN                                                        SEMAN347
 1400 CONTINUE
      LVVTYP = 3
      LVVPOS = 1
      LVINDX = 0
      LVFUNC=          STJ
      LVVARG=            R
      CALL LVFIND(LVINDX,LVINDX,LVINDX,LVINDX)
      LV1     AA8 =             R
      IF (LVVAL.NE.-1) LV1     AA8 = LVVAL
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.EQ.-1) GO TO           11
                R = LV1     AA8
C  LEFT PAREN FOUND IN I/O LIST                                     SEMAN349
      NFLAG=NFLAG+1                                                 SEMAN350
      FL(NFLAG)=MARGS                                               SEMAN351
      RETURN                                                        SEMAN352
 1500 CONTINUE
      LVVTYP = 3
      LVVPOS = 1
      LVINDX = 0
      LVFUNC=          STJ
      LVVARG=            R
      CALL LVFIND(LVINDX,LVINDX,LVINDX,LVINDX)
      LV1     AA8 =             R
      IF (LVVAL.NE.-1) LV1     AA8 = LVVAL
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.EQ.-1) GO TO           11
                R = LV1     AA8
      IF(STJ .EQ. COMMA) RETURN                                     SEMAN354
C  RIGHT PAREN FOUND IN I/O LIST                                    SEMAN355
      NFLAG=NFLAG-1                                                 SEMAN356
      RETURN                                                        SEMAN357
 1600 CONTINUE                                                      SEMAN358
      TYPE1=4                                                       SEMAN359
      RETURN                                                        CY58B  7
 1610 CALL ERROR(95)                                                CY58B  8
      STOP                                                          CY58B  9
      RETURN
25000 CONTINUE
      LV2       A=LV2       B=LV2       C=LV2       D=0
      LV2       E=LV2       F=LV2       G=LV2       H=0
      LV2       I=LV2       J=LV2       K=LV2       L=0
      LV2       M=LV2       N=LV2       O=LV2       P=0
      LV2       Q=LV2       R=LV2       S=LV2       T=0
      LV2       U=LV2       V=LV2       W=LV2       X=0
      LV2       Y=LV2       Z=LV2       0=LV2       1=0
      LV2       2=LV2       3=LV2       4=LV2       5=0
      LV2       6=LV2       7=LV2       8=LV2       9=0
      LV2      AA=LV2      AB=LV2      AC=LV2      AD=0
      LV2      AE=LV2      AF=LV2      AG=LV2      AH=0
      LV2      AI=LV2      AJ=LV2      AK=LV2      AL=0
      LV2      AM=LV2      AN=LV2      AO=LV2      AP=0
      LV2      AQ=LV2      AR=LV2      AS=LV2      AT=0
      LV2      AU=LV2      AV=LV2      AW=LV2      AX=0
      LV2      AY=LV2      A7=LV2      A0=LV2      A1=0
      LV2      A2=LV2      A3=LV2      A4=LV2      A5=0
      LV2      A6=LV2      A7=LV2      A8=LV2      A9=0
      LV2      BA=LV2      BB=LV2      BC=LV2      BD=0
      LV2      BE=LV2      BF=LV2      BG=LV2      BH=0
      LV2      BI=LV2      BJ=LV2      BK=LV2      BL=0
      LV2      BM=LV2      BN=LV2      B0=LV2      BP=0
      LV2      BQ=LV2      BR=LV2      BS=LV2      BT=0
      LV2      BU=LV2      BV=LV2      BW=LV2      BX=0
      LV2      BY=LV2      B7=LV2      B0=LV2      B1=0
      LV2      B2=LV2      B3=LV2      B4=LV2      B5=0
      GO TO 25001
      END
```

GIRL Version

```
$       SUBROUTINE SEMANT(N,FAIL)                                          SEMANT     2
        COMMON/FUNC/  NARY(5,12),MARGS,IARGS(50),FNCLOC(5),NFUNC           CY588      3
        COMMON/HL/HOL,ACTION,FUNC1,FUNC2,FUNC3,LEFT,RIGHT,STRING,MAXJ      SEMANT     4
        COMMON /TYP/ NARRAY,TYPE1,TYPE2,ERRFLG                             SEMANT     5
        COMMON /STRING/ NTYPE,NSTR,STR                                     SEMANT     6
        COMMON /JL/ JSTOP                                                  SEMANT     7
        COMMON /GIRL/NTERMS,PLUS,MINUS,SLASH,LPAR,RPAR,COMMA,STAR,EXP,LT,  SEMANT     8
       +LE,GT,GE,EQ,NE,OR,AND,NOT,EQUALS,OPRAND                            SEMANT     9
        COMMON/NEEDS/STJ,JSTACK,R,JAS,J,JLAST,RTEMP,STACK(400)             SEMANT    10
        COMMON /NEED/ START,ASSOC,LEVEL,STOP                               SEMANT    11
        COMMON/NOPAR/NOPAR,NDEP,NDEPTH,NFLAG                               SEMANT    12
        INTEGER HOL,ACTION,FUNC,LEFT,RIGHT,STRING,RPAR,STJ,R,STACK         SEMANT    13
       +,EXP,FUNC1,FUNC2,FUNC3,TYPE1,TYPE2,TYPE(5),STR(1),STOP             SEMANT    14
       $ ,ALPHA,BETA,GAMMA,OPRAND,EQUALS,AND,OR,COMMA                      SEMANT    15
        LOGICAL SKIP,FLAG,ERRFLG,FAIL,NOTFLG                              SEMANT    16
        INTEGER FUNCRF,ZERO,BITPUT,PLUS,FL(3),BITGET                       SEMANT    17
        INTEGER GETTYP,GETDIM                                              SEMANT    18
        DATA FLAG/.FALSE./,FUNCRF/86/,ZERO/0/                              SEMANT    19
        DATA (TYPE(I),I=1,5)/4HREAL,6HCOMPLX,6HDOUBLE,6HINTEGR,6HLOGICL/   SEMANT    20
        GETTYP(II)=MOD(II,100000)/10000                                    SEMANT    21
        GETDIM(II)=MOD(II,1000000)/100000                                  SEMANT    22
G       EXECUTE                                                            SEMANT    23
        FAIL=.FALSE.                                                       SEMANT    24
        IF(N .EQ. 0) GO TO 999                                             SEMANT    25
        GO TO(10,20,30,40,50,60,70,80,90,1000,1100,1200,1300,1400,1500,    SEMANT    26
       $ 1600),N                                                           SEMANT    27
G    10 R=STJ/11 *R//12                                                    SEMANT    28
     11 FAIL=.TRUE.                                                        SEMANT    29
        RETURN                                                             SEMANT    30
C PRIMARY RECOGNIZED                                                       SEMANT    31
     12 IF(STJ .EQ. PLUS .OR. STJ .EQ. MINUS) GO TO 126                    SEMANT    32
        IF(STJ .NE. RPAR) GO TO 121                                        SEMANT    33
        JSTACK=JSTACK+1                                                    SEMANT    34
        STACK(JSTACK)=SHIFT(STOP,45) .OR. SHIFT(J,15)                      SEMANT    35
        NTMP=R                                                             SEMANT    36
        CALL SLEVEL(SKIP)                                                  SEMANT    37
        JSTACK=JSTACK-1                                                    SEMANT    38
        R=NTMP                                                             SEMANT    39
        JLAST=1                                                            SEMANT    40
        IF(JSTOP .GT. 0) JLAST=BITGET(STACK(JSTOP),45,15)                  SEMANT    41
G       STRING=HOL.JLAST(-STRING,STRING ''TYPE1'')                         SEMANT    42
        RETURN                                                             SEMANT    43
  121 CONTINUE                                                             SEMANT    44
C  GET TYPE                                                               SEMANT    45
        BETA=GETDIM(STR(J))                                                SEMANT    46
        IF(BETA .NE. 5) GO TO 125                                          SEMANT    47
C  OPERAND IS A FUNCTION REFERENCE                                         SEMANT    48
        IF(NDEP .EQ. 0) GO TO 18                                           SEMANT    49
G       OPRAND FUNC1-.-1 ''1''                                             SEMANT    50
     18 R=FUNCRF                                                           SEMANT    51
        JSTACK=JSTACK+1                                                    SEMANT    52
        STACK(JSTACK)=SHIFT(R,45) .OR. SHIFT(J+1,15)                       SEMANT    53
  125 ALPHA=GETTYP(STR(J))                                                 SEMANT    54
        IF(TYPE1 .GE. 0) GO TO 13                                          SEMANT    55
C  SET TYPE OF STATEMENT                                                   SEMANT    56
        TYPE1=ALPHA                                                        SEMANT    57
        IF(NTYPE .EQ. 3) TYPE1=-1                                          SEMANT    58
```

```
G 126 STRING+HOL.J(-STRING,STRING ''TYPE1'')                                    SEMANT     59
      RETURN                                                                    SEMANT     60
   13 IF(FLAG) GO TO 15                                                         SEMANT     61
C  CHECK FOR MIXED MODE EXPRESSION                                             SEMANT     62
      IF(TYPE1 .EQ. ALPHA .OR. ALPHA .EQ. 5) GO TO 16                           CY60        1
      N1=TYPE1+1                                                                SEMANT     64
      N2=ALPHA+1                                                                SEMANT     65
      ERRFLG=.TRUE.                                                             SEMANT     66
      CALL ERROR(77,TYPE(N1),TYPE(N2))                                          SEMANT     67
G     STRING+HOL.J(-STRING,STRING ''TYPE1'')                                    SEMANT     68
      RETURN                                                                    SEMANT     69
C  PARSING  AN EXPONENT                                                         SEMANT     70
   15 IF(ALPHA .EQ. 3 .AND. TYPE1 .EQ. 3) GO TO 16                              SEMANT     71
      IF((TYPE1 .EQ. 0 .OR. TYPE1 .EQ. 2) .AND. (ALPHA .EQ. 0 .OR.              SEMANT     72
     $ ALPHA .EQ. 2)) GO TO 16                                                  SEMANT     73
      CALL ERROR(78,J)                                                          SEMANT     74
      ERRFLG=.TRUE.                                                             SEMANT     75
G     STRING+HOL.J(-STRING,STRING ''TYPE1'')                                    SEMANT     76
      RETURN                                                                    SEMANT     77
   16 IF((.NOT. FLAG .AND. TYPE1 .LT. ALPHA).OR.(FLAG .AND. ALPHA .NE. 3       SEMANT     78
     + .AND. TYPE1 .LT. ALPHA)) TYPE1=ALPHA                                     SEMANT     79
G     STRING+HOL.J(-STRING,STRING ''TYPE1'')                                    SEMANT     80
      RETURN                                                                    SEMANT     81
C  WILL SCAN AN EXPONENT                                                       SEMANT     82
   20 IF(STJ .LT. 0) RETURN                                                     SEMANT     83
G     R+STJ/11 'R                                                               SEMANT     84
      FLAG=.TRUE.                                                               SEMANT     85
      RETURN                                                                    SEMANT     86
C  RECOGNIZED A TERM,PRODUCT OR PRIMARY PERHAPS NEEDING PARENTHESIZATION        SEMANT     87
   30 CONTINUE                                                                  SEMANT     88
      KTMP=R                                                                    SEMANT     89
      IF(NDEP .EQ. 0) GO TO 34                                                  SEMANT     90
G     OPRAND FUNC1-.-1 ''1''                                                    SEMANT     91
   34 CONTINUE                                                                  SEMANT     92
      ITEST=0                                                                   SEMANT     93
      IF(STJ .LT. 0) GO TO 31                                                   SEMANT     94
G     R+STJ 'R//32                                                              SEMANT     95
G  31 R+STOP/39 'R                                                              SEMANT     96
      IF(STJ .LT. 0 .AND. KTMP .EQ. 889) GO TO 38                               SEMANT     97
      ITEST=-1                                                                  SEMANT     98
   32 CONTINUE                                                                  SEMANT     99
C IF UNARY PLUS OR MINUS RETURN                                                 SEMANT    100
      ISTCK=BITGET(STACK(JSTOP),15,15)                                          SEMANT    101
      IF(ISTCK .NE. 288 .AND. ISTCK .NE. 110) GO TO 33                          SEMANT    102
      JLAST=BITGET(STACK(JSTOP),45,15)-1                                        SEMANT    103
      IF(ISTCK .EQ. 288) JLAST=JLAST-1                                          SEMANT    104
G     STRING+HOL.JLAST(-STRING,STRING ''TYPE1'')                                SEMANT    105
G     STRING+HOL.J(-STRING,STRING ''TYPE1'')                                    SEMANT    106
      IF(ITEST .LT. 0) J=J-1                                                    SEMANT    107
      RETURN                                                                    SEMANT    108
   33 CONTINUE                                                                  SEMANT    109
      IF(ITEST .LT. 0) J=J-1                                                    SEMANT    110
      JSTACK=JSTACK+1                                                           SEMANT    111
      STACK(JSTACK)=SHIFT(STOP,45) .OR. SHIFT(J,15)                             SEMANT    112
      NTMP=R                                                                    SEMANT    113
      CALL SLEVEL(SKIP)                                                         SEMANT    114
      JSTACK=JSTACK-1                                                           SEMANT    115
```

```
      R=NTMP                                                                    SEMANT    116
      JLAST=1                                                                   SEMANT    117
      IF(JSTOP .GT. 0) JLAST=BITGET(STACK(JSTOP),45,15)                         SEMANT    118
      NTMP=TYPE1                                                                SEMANT    119
      JJ=JLAST                                                                  SEMANT    120
      IF(BITGET(STACK(JSTACK),15,15) .EQ. 418 .AND. JLAST .GT. 1)               SEMANT    121
     $ JJ=JLAST-1                                                               SEMANT    122
G     STRING+HOL.JJ+STRING 'TYPE1                                               SEMANT    123
      IF(.NOT. FLAG .OR. NTMP .EQ. 3) GO TO 35                                  SEMANT    124
      IF((NTMP.EQ.0.OR.NTMP.EQ.2).AND.(TYPE1.EQ.0.OR.TYPE1.EQ.2))GOTO 35        SEMANT    125
      ERRFLG=.TRUE.                                                             SEMANT    126
      CALL ERROR(78,J)                                                          SEMANT    127
   35 CONTINUE                                                                  SEMANT    128
      IF(TYPE1 .GT. 2 .OR. NTYPE .GT. 1) GO TO 38                               SEMANT    129
      FUNC=FUNC1                                                                SEMANT    130
      IF(TYPE1 .EQ. 1) FUNC=FUNC2                                               SEMANT    131
      IF(TYPE1 .EQ. 2) FUNC=FUNC3                                               SEMANT    132
G     STRING+HOL.JLAST LEFT(.1 LPAR,.1 FUNC)                                    SEMANT    133
G     STRING+HOL.J RIGHT RPAR                                                   SEMANT    134
   38 IF(ITEST .LT. 0 .AND. STJ .LT. 0) J=J+1                                   SEMANT    135
      FLAG=.FALSE.                                                              SEMANT    136
      RETURN                                                                    SEMANT    137
   39 FLAG=.FALSE.                                                              SEMANT    138
      GO TO 11                                                                  SEMANT    139
C  CHECK FOR CORRECTNESS OF SUBSCRIPTS AND DO IMPLIED LIST PARAMETERS           SEMANT    140
   40 NR=R                                                                      SEMANT    141
G     R+STJ/11 *R                                                               SEMANT    142
      NBETA=GETDIM(STR(J))                                                      SEMANT    143
      IF(NR .EQ. 359 .AND. NBETA .NE. 4) GO TO 47                               SEMANT    144
      ALPHA=GETTYP(STR(J))                                                      SEMANT    145
      GAMMA=STR(J)/1000000                                                      SEMANT    146
      IF(NTYPE .EQ. 3) GO TO 45                                                 SEMANT    147
      IF(NR .EQ. 839 .AND. NBETA .EQ. 0) GO TO 45                               SEMANT    148
      IF(NR .EQ. 359) GO TO 45                                                  SEMANT    149
      IF(NR .EQ. 21 .AND. NBETA .EQ. 4) GO TO 45                                SEMANT    150
      IF(NBETA .EQ. 0 .AND. NR .EQ. 935) GO TO 45                               SEMANT    151
      IF(NTYPE .EQ. 2 .AND. NR .EQ. 935 .AND. STR(J-1) .NE. -7) GO TO 45        SEMANT    152
      IF(NTYPE .EQ. 2 .AND. NR .EQ. 359 .AND. NBETA .EQ. 0) GO TO 11            SEMANT    153
      CALL ERROR(79,J)                                                          SEMANT    154
      ERRFLG=.TRUE.                                                             SEMANT    155
   45 CONTINUE                                                                  SEMANT    156
      IF(GAMMA .GE. 6 .AND. NBETA .EQ. 4) CALL ERROR(76)                        SEMANT    157
      IF(ALPHA .EQ. 3) GO TO 46                                                 SEMANT    158
      N1=ALPHA+1                                                                SEMANT    159
      ERRFLG=.TRUE.                                                             SEMANT    160
      CALL ERROR(80,TYPE(N1),J)                                                 SEMANT    161
   46 IF(NBETA .EQ. 4) RETURN                                                   SEMANT    162
      MARGS=MARGS+1                                                             SEMANT    163
G     OPRAND(FUNC2 ''MARGS'',FUNC3 ''J'',LEVEL ''NDEPTH'')                     SEMANT    164
      IVR=(MARGS+2)/3                                                           SEMANT    165
      IF(IVR .GT. 50) GO TO 1610                                                CY588       5
      ICOL=20*MOD(MARGS-1,3)+10                                                 SEMANT    166
      IVAL=MOD(STR(J),10000)                                                    SEMANT    167
      IARGS(IVR)=BITPUT(IARGS(IVR),IVAL,ICOL)                                   SEMANT    168
      IF(NR .EQ. 839) GO TO 49                                                  SEMANT    169
      IF(NTYPE .NE. 3) RETURN                                                   SEMANT    170
C  FLAG SUBSCRIPT IN I/O LIST                                                   SEMANT    171
```

```
      IARGS(IVR)=BITPUT(IARGS(IVR),1,ICOL+5)                                SEMANT   172
      RETURN                                                                SEMANT   173
C FLAG DO INDEX IN I/O LIST                                                 SEMANT   174
   49 IARGS(IVR)=BITPUT(IARGS(IVR),2,ICOL+5)                                SEMANT   175
      IF(NFLAG .LT. 1) RETURN                                               SEMANT   176
      IARGS(IVR)=BITPUT(IARGS(IVR),FL(NFLAG),ICOL+10)                       SEMANT   177
      RETURN                                                                SEMANT   178
   47 NR=R                                                                  SEMANT   179
C   SUBSCRIPT DOES NOT BEGIN WITH CONSTANT, FORCE SEARCH FOR VARIABLE       SEMANT   180
      GO TO 11                                                              SEMANT   181
C  CHECK FOR PROPER NUMBER OF SUBSCRIPTS                                    SEMANT   182
   50 IF( BETA .EQ. 4 .OR. R .NE. 452) GO TO 52                             SEMANT   183
      MARGS=MARGS+1                                                         SEMANT   184
G     OPRAND(FUNC2 ''MARGS'',FUNC3 ''J-1'',LEVEL ''NDEPTH'')                SEMANT   185
      IVR=(MARGS+2)/3                                                       SEMANT   186
      IF(IVR .GT. 50) GO TO 1610                                            CY588      6
      ICOL=20*MOD(MARGS-1,3)+10                                             SEMANT   187
      IVAL=MOD(STR(J-1),10000)                                              SEMANT   188
      IARGS(IVR)=BITPUT(IARGS(IVR),IVAL,ICOL)                               SEMANT   189
      IF(NOPAR .LE. 0) GO TO 52                                             SEMANT   190
G     OPRAND+ACTION.-1/52 *MFUNC                                            SEMANT   191
      IARGS(IVR)=BITPUT(IARGS(IVR),MFUNC ,ICOL+3)                           SEMANT   192
      IARGS(IVR)=BITPUT(IARGS(IVR),NARGS,ICOL+9)                            SEMANT   193
C IF NO STRING LEFT, RETURN IF CONSTANT,VARIABLE OR I/O LIST                SEMANT   194
   52 IF(STJ .LT. 0 .AND. (BETA .EQ. 0 .OR. BETA .EQ. 4                     SEMANT   195
     $ .OR. NTYPE .EQ. 3)) RETURN                                           SEMANT   196
      IF(BETA .GT. NARRAY) GO TO 55                                         SEMANT   197
      ERRFLG=.TRUE.                                                         SEMANT   198
      CALL ERROR(81,J)                                                      SEMANT   199
   55 IF(R .EQ. 452) NARRAY=0                                               SEMANT   200
      IF(R .EQ. 318) NARRAY=1                                               SEMANT   201
      IF(R .EQ.  60) NARRAY=2                                               SEMANT   202
      IF(R .EQ. 103) NARRAY=3                                               SEMANT   203
      IF(STJ .LT. 0) GO TO 58                                               SEMANT   204
G     R+STJ *R//56                                                          SEMANT   205
   58 IF(NTYPE .EQ. 3 .AND. NARRAY .EQ. 0) GO TO 57                         SEMANT   206
      IF(BETA .GE. 1 .AND. BETA .LE. 3 .AND. NOPAR .EQ. 0)                  SEMANT   207
     * CALL ERROR(82,J)                                                    SEMANT   208
   57 NARRAY=-1                                                             SEMANT   209
      GO TO 11                                                              SEMANT   210
   56 IF(NTYPE .EQ. 3 .AND. NARRAY .EQ. 0) RETURN                           SEMANT   211
      IF(STJ .EQ. RPAR .AND. NARRAY .LT.  BETA .AND. J .EQ. MAXJ)           SEMANT   212
     $ CALL ERROR(82,J)                                                     SEMANT   213
      IF(STJ .EQ. RPAR) NARRAY=-1                                           SEMANT   214
      RETURN                                                                SEMANT   215
C  RESET TYPE OF STATEMENT IN ANTICIPATION OF SEARCH FOR BOOLEAN PRIMARY     SEMANT   216
   60 CONTINUE                                                              SEMANT   217
      NOTFLG=.FALSE.                                                        SEMANT   218
      IF(STR(J-1) .EQ. -17) NOTFLG=.TRUE.                                   SEMANT   219
      TYPE1=-1                                                              SEMANT   220
G     STRING+HOL.J(-STRING,STRING ''TYPE1'')                                SEMANT   221
      IF(STJ .NE. OPRAND) GO TO 65                                          SEMANT   222
      ALPHA=GETTYP(STR(J))                                                  SEMANT   223
      BETA=GETDIM(STR(J))                                                   SEMANT   224
      IF(ALPHA .NE. 4) GO TO 11                                             SEMANT   225
   65 CONTINUE                                                              SEMANT   226
G     R+STJ/11 *R                                                           SEMANT   227
```

```
      RETURN                                                                SEMANT   228
C  IF BOOLEAN PRIMARY IS AN ARETHMETIC COMPARE CONTINUE PARSING PRIMARY     SEMANT   229
   70 IF(STJ .LT. 0) RETURN                                                 SEMANT   230
      IF(TYPE1 .EQ. 4) GO TO 75                                             SEMANT   231
G     R+STJ 'R/11                                                           SEMANT   232
C  RELATIONAL OPERATOR FOUND                                                SEMANT   233
      IF(NDEP .EQ. 0) RETURN                                                SEMANT   234
G     OPRAND FUNC1-.-1 ''1''                                                SEMANT   235
      RETURN                                                                SEMANT   236
C  IF BOOLEAN VARIABLE OR CONSTANT, SET STATE TO STOP                       SEMANT   237
   75 R=STOP                                                                SEMANT   238
      JSTACK=JSTACK+1                                                       SEMANT   239
      STACK(JSTACK)=SHIFT(R,45) .OR. SHIFT(J,15)                            SEMANT   240
      GO TO 11                                                              SEMANT   241
C  COMPARE TYPES ON BOTH SIDES OF RELATIONAL EXPRESSION                     SEMANT   242
   80 IF(TYPE1 .EQ. 0 .OR. TYPE1 .EQ. 2 .OR. TYPE1 .EQ. 3) GO TO 85         SEMANT   243
      ERRFLG=.TRUE.                                                         SEMANT   244
      CALL ERROR(83,J)                                                      SEMANT   245
      TYPE1=-1                                                              SEMANT   246
G     STRING+HOL.J(-STRING,STRING ''TYPE1'')                                SEMANT   247
   85 TYPE2=TYPE1                                                           SEMANT   248
      GO TO 11                                                              SEMANT   249
C  BOOLEAN PRIMARY RECOGNIZED-SET TYPE TO BOOLEAN AND CONTINUE PARSE        SEMANT   250
   90 IF(TYPE1 .EQ. TYPE2 .OR. TYPE1+TYPE2 .EQ. 2 .OR.                      SEMANT   251
     + TYPE2 .LT. 0) GO TO 95                                               SEMANT   252
      N1=TYPE1+1                                                            SEMANT   253
      N2=TYPE2+1                                                            SEMANT   254
      CALL ERROR(77,TYPE(N1),TYPE(N2))                                      SEMANT   255
      ERRFLG=.TRUE.                                                         SEMANT   256
   95 TYPE1=4                                                               SEMANT   257
      TYPE2=-1                                                              SEMANT   258
      IF(STJ .LT. 0) RETURN                                                 SEMANT   259
G     STRING+HOL.J(-STRING,STRING ''TYPE1'')                                SEMANT   260
      GO TO 11                                                              SEMANT   261
C  PARSE REACHED BLIND ALLEY-MUST BACK UP AND REMOVE PARENTHESES CREATED    SEMANT   262
  999 JM=BITGET(STACK(JSTACK),45,15)                                        SEMANT   263
      K=JM                                                                  SEMANT   264
      DO 996 KK=JM,J                                                        SEMANT   265
G     STRING+HOL.K/995+STRING.1/996 'TYPE1                                  SEMANT   266
      GO TO 995                                                             SEMANT   267
  996 K=K-1                                                                 SEMANT   268
  995 CONTINUE                                                              SEMANT   269
      DO 998 I=JM,J                                                         SEMANT   270
G     STRING+HOL.I(-LEFT,-RIGHT)                                            SEMANT   271
  998 CONTINUE                                                              SEMANT   272
G 980 OPRAND+FUNC3.-1 'JN                                                   SEMANT   273
      IF(JN .LT. JM) GO TO 985                                              SEMANT   274
G     OPRAND(+FUNC2-.-1,+FUNC3-.-1,+LEVEL-.-1)                              SEMANT   275
      GO TO 980                                                             SEMANT   276
G 985 OPRAND+FUNC2.-1 'MARGS                                                SEMANT   277
G     OPRAND+LEVEL.-1 'NDEPTH                                               SEMANT   278
      RETURN                                                                SEMANT   279
                                                                            SEMANT   280
C   RECOGNIZED FUNCTION-PREPARE TO SET TYPE OF ARGUMENTS FOR THE 'NDEPTH    SEMANT   281
C           FUNCTION IN THIS STMT                                           SEMANT   282
 1000 CONTINUE                                                              SEMANT   283
      NDEPTH=NDEPTH+1                                                       SEMANT   284
```

```
      NDEP=NDEP+1                                                          SEMANT   285
      NARGS=0                                                              SEMANT   286
G     R+STJ/11 'R                                                          SEMANT   287
      NARGS=1                                                              SEMANT   288
G     OPRAND(OPRAND ''TYPE1'',STRING''NARGS'',ACTION ''NDEPTH'')           SEMANT   289
G     OPRAND FUNC1 ''0''                                                   SEMANT   290
      TYPE1=-1                                                             SEMANT   291
      NOPAR=NOPAR+1                                                        SEMANT   292
      RETURN                                                               SEMANT   293
                                                                           SEMANT   294
C KEEP TRACK OF THE NUMBER AND TYPES OF ARGUMENTS IN FUNCTION CALLS        SEMANT   295
C   MUST USE STACK FOR POSSIBLE RECURSIVE FUNCTION USE                     SEMANT   296
 1100 CONTINUE                                                             SEMANT   297
G     R+STJ/11 'R                                                          SEMANT   298
G     OPRAND(+STRING.-1/1103 'NARGS,+STRING-.-1,+ACTION.-1 'MFUNC)         SEMANT   299
G     OPRAND+FUNC1.-1 'IEXP                                                SEMANT   300
 1103 CONTINUE                                                             SEMANT   301
C  STORE ARGUMENT TYPES                                                    SEMANT   302
      IF(NDEPTH .GT. 5) CALL ERROR(85)                                     SEMANT   303
      IF(NDEPTH .GT. 5) GO TO 1130                                         SEMANT   304
      IF(NARGS .LE. 63) GO TO 1104                                         SEMANT   305
      ERRFLG=.TRUE.                                                        SEMANT   306
      CALL ERROR(84,NDEPTH)                                                SEMANT   307
      GO TO 11                                                             SEMANT   308
 1104 CONTINUE                                                             SEMANT   309
      MM=(11+NARGS)/6                                                      SEMANT   310
      ITEMP=NARGS-6*(MM-2)                                                 SEMANT   311
      ICOL=9*ITEMP-6                                                       SEMANT   312
      NARY(MFUNC,MM)=BITPUT(NARY(MFUNC,MM),MOD((TYPE1+1),6),ICOL)          CY60       2
      IF(STR(J-2) .NE. -6 .AND. STR(J-2) .NE. -4) GO TO 1130               SEMANT   314
      NDIM=GETDIM(STR(J-1))                                                SEMANT   315
      IF(NDIM .GE. 4) GO TO 1130                                           SEMANT   316
C  STORE DIMENSIONALITY OF ARGUMENTS                                       SEMANT   317
      NARY(MFUNC ,MM)=BITPUT(NARY(MFUNC ,MM),NDIM,ICOL+3)                  SEMANT   318
 1130 CONTINUE                                                             SEMANT   319
      NARY(MFUNC ,MM)=BITPUT(NARY(MFUNC ,MM),IEXP,54+ITEMP)                SEMANT   320
      IF(STJ .EQ. COMMA) GO TO 1105                                        SEMANT   321
      NARY(MFUNC,1)=NARGS                                                  SEMANT   322
G     OPRAND(+ACTION-.-1,+OPRAND.-1/1135 'TYPE1)                           SEMANT   323
G     OPRAND(+FUNC1-.-1,+OPRAND-.-1)                                       SEMANT   324
 1135 NOPAR=NOPAR-1                                                        SEMANT   325
      NDEP=NDEP-1                                                          SEMANT   326
      RETURN                                                               SEMANT   327
 1105 TYPE1=-1                                                             SEMANT   328
G     STRING+HOL.J(-STRING,STRING ''TYPE1'')                               SEMANT   329
      NARGS=NARGS+1                                                        SEMANT   330
G     OPRAND STRING ''NARGS''                                              SEMANT   331
G     OPRAND FUNC1-.-1 ''0''                                               SEMANT   332
      RETURN                                                               SEMANT   333
C  SAVE TYPE OF STAEMENT WHILE PARSING EXPONENT                            SEMANT   334
 1200 CONTINUE                                                             SEMANT   335
G     STRING+HOL.J(-STRING,STRING ''TYPE1'')                               SEMANT   336
      TYPE1=-1                                                             SEMANT   337
G     R+STJ/11 'R                                                          SEMANT   338
      RETURN                                                               SEMANT   339
 1300 CONTINUE                                                             SEMANT   340
      IF(STJ .LT. 0) RETURN                                                SEMANT   341
```

```
      IF(STJ .NE. AND .AND. STJ .NE. OR .AND. STJ .NE. NOT) GO TO 11      SEMANT   342
G     R+STJ/11 *R                                                         SEMANT   343
      IF(NOEP .EQ. 0) RETURN                                              SEMANT   344
C  LOGICAL OPERATOR FOUND                                                 SEMANT   345
G     OPRAND FUNC1-.-1 **1**                                              SEMANT   346
      RETURN                                                              SEMANT   347
G1400 R+STJ/11 *R                                                         SEMANT   348
C  LEFT PAREN FOUND IN I/O LIST                                           SEMANT   349
      NFLAG=NFLAG+1                                                       SEMANT   350
      FL(NFLAG)=MARGS                                                     SEMANT   351
      RETURN                                                              SEMANT   352
G1500 R+STJ/11 *R                                                         SEMANT   353
      IF(STJ .EQ. COMMA) RETURN                                           SEMANT   354
C  RIGHT PAREN FOUND IN I/O LIST                                          SEMANT   355
      NFLAG=NFLAG-1                                                       SEMANT   356
      RETURN                                                              SEMANT   357
 1600 CONTINUE                                                            SEMANT   358
      TYPE1=4                                                             SEMANT   359
      RETURN                                                              CY588      7
 1610 CALL ERROR(95)                                                      CY588      8
      STOP                                                                CY588      9
G     COMPLETE                                                            SEMANT   360
```

```
      SUBROUTINE SEPAR                                                  SEPAR      2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH       2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A     80
     2 LTYP,ITYP,IBLKOT,MODE,IERR,IDES                                  RICH       4
      COMMON/FORMAT/IDESST,IDESNO,IGPST,IGPND,IGRP,SEPST,SEPND,         CY58A      4
     1 DIR,ICOM,ISEP                                                    SEPAR      5
      INTEGER A,SEPST,SEPND,DIR,BLANK,SLASH,COMMA                       SEPAR      6
      DATA BLANK/1H /,SLASH/1H//,COMMA/1H,/                             SEPAR      7
C** THIS ROUTINE CHECKS THE SYNTAX OF FIELD SEPARATORS AND RETURNS      SEPAR      8
C**      ISEP=1 - VALID                                                 SEPAR      9
C**      ISEP=0 - NON-SEPARATOR                                         SEPAR     10
C**      ISEP=-1 - INVALID                                              SEPAR     11
      ICOM=0                                                            SEPAR     12
      ISLASH=0                                                          SEPAR     13
      DO 20 I=1,N                                                       SEPAR     14
      JJ=SEPST+DIR*(I-1)                                                SEPAR     15
      IF(A(JJ) .EQ. BLANK) GO TO 20                                     SEPAR     16
      IF(A(JJ) .EQ. SLASH) GO TO 5                                      SEPAR     17
      IF(A(JJ) .EQ. COMMA) GO TO 10                                     SEPAR     18
      GO TO 30                                                          SEPAR     19
    5 CONTINUE                                                          SEPAR     20
      ISLASH=1                                                          SEPAR     21
      IF(ICOM .EQ. 1) GO TO 40                                          SEPAR     22
      GO TO 20                                                          SEPAR     23
   10 IF(ISLASH .EQ. 1 .OR. ICOM .EQ. 1) GO TO 40                       SEPAR     24
      ICOM=1                                                            SEPAR     25
   20 CONTINUE                                                          SEPAR     26
      GO TO 40                                                          SEPAR     27
   30 IF(ISLASH .EQ. 0 .AND. ICOM .EQ. 0) GO TO 35                      SEPAR     28
      ISEP=1                                                            SEPAR     29
      SEPND=JJ-DIR                                                      SEPAR     30
      RETURN                                                            SEPAR     31
   35 CONTINUE                                                          SEPAR     32
      ISEP=0                                                            SEPAR     33
      SEPND=JJ                                                          SEPAR     34
      RETURN                                                            SEPAR     35
   40 ISEP=-1                                                           SEPAR     36
      RETURN                                                            SEPAR     37
      END                                                               SEPAR     38
```

```
      SUBROUTINE SIMP                                                    SIMP     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),   RICH     2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,         CY58A   80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                   RICH     4
      COMMON/BASBLK/IBLOCK(2500),NBLOCK,NB,NBRNCH                        CY58A    6
      DIMENSION IALPH1(7),IALPH2(9),IALPH3(5),IALPH4(10)                 SIMP     5
      DATA (IALPH1(I),I=1,7)/1HR,1HE,1HT,1HU,1HR,1HN,1H /,               SIMP     6
     1 (IALPH2(I),I=1,9)/1HC,1HO,1HN,1HT,1HI,1HN,1HU,1HE,1H /,          SIMP     7
     2 (IALPH3(I),I=1,5)/1HS,1HT,1HO,1HP,1H /                           SIMP     8
     3 ,(IALPH4(I),I=1,10)/1HB,1HL,1HO,1HC,1HK,1HD,1HA,1HT,1HA,1H /     SIMP     9
C** THIS ROUTINE PROCESSES RETURN,CONTINUE,STOP, AND BLOCK DATA          SIMP    10
      IF(ITYP .EQ. 10) GO TO 25                                          SIMP    11
      IF(ITYP .EQ. 7) GOTO 15                                            SIMP    12
      IF(ITYP .EQ. 29) GO TO 35                                          SIMP    13
C** CHECK RETURN STATEMENT AND STORE BRANCH IN BASIC BLOCK TABLE          SIMP    14
      DO 10 I=1,7                                                        SIMP    15
      IF(NEXT(JPTR) .NE. IALPH1(I)) GO TO 50                             SIMP    16
   10 CONTINUE                                                           SIMP    17
      NB=1                                                               SIMP    18
      NBRNCH=1                                                           SIMP    19
      NBLOCK=NBLOCK+1                                                    SIMP    20
      IBLOCK(NBLOCK)=999                                                 SIMP    21
      RETURN                                                             SIMP    22
C** CHECK CONTINUE STATEMENT                                             SIMP    23
   15 DO 20 I=1,9                                                        SIMP    24
      IF(NEXT(JPTR) .NE. IALPH2(I)) GO TO 50                             SIMP    25
   20 CONTINUE                                                           SIMP    26
      RETURN                                                             SIMP    27
C** CHECK STOP STATEMENT AND STORE BRANCH IN BASIC BLOCK TABLE           SIMP    28
   25 DO 30 I=1,5                                                        SIMP    29
      IF(NEXT(JPTR) .NE. IALPH3(I)) GO TO 50                             SIMP    30
   30 CONTINUE                                                           SIMP    31
      NB=1                                                               SIMP    32
      NBRNCH=1                                                           SIMP    33
      NBLOCK=NBLOCK+1                                                    SIMP    34
      IBLOCK(NBLOCK)=999                                                 SIMP    35
      RETURN                                                             SIMP    36
C** CHECK BLOCK DATA STATEMENT                                           SIMP    37
   35 DO 40 I=1,10                                                       SIMP    38
      IF(NEXT(JPTR) .NE. IALPH4(I)) GO TO 50                             SIMP    39
   40 CONTINUE                                                           SIMP    40
      RETURN                                                             SIMP    41
   50 CALL ERROR(7)                                                      SIMP    42
      RETURN                                                             SIMP    43
      END                                                                SIMP    44
```

FORTRAN Version

```
      SUBROUTINE SLEVEL(FAIL)                                                SLEVEL 2
      COMMON/LVARGS/LVFUNC,LVVARG,LVVAD,LVVPOS,LVVTYP,       LVVAL,
     +LVHEAD,LVVNVL,LVDEST,LVVALS(10),LVTYPE(10),LVSKIP
      COMMON/LVTABL/LVTSIZ,LVMAP(    1)/LVVSEQ/LVSIZE,LVSQSP(     1)
      COMMON /NEED/ START,ASSOC,LEVEL,STOP                                   SLEVEL 3
      COMMON/NEEDS/STJ,JSTACK,R,JAS,J,JLAST,RTEMP,STACK(400)                 SLEVEL 4
      COMMON /STRING/ NNN(2),STR                                             SLEVEL 5
      COMMON /JL/ JSTOP                                                      SLEVEL 6
      INTEGER START,STOP,ASSOC,STACK,STR(1),STJ,R,RTEMP                      SLEVEL 7
      INTEGER BITGET                                                         SLEVEL 8
      LOGICAL FAIL                                                           SLEVEL 9
      GO TO 25000
25001 CONTINUE
      RTEMP=0                                                                SLEVEL11
      JSTOP=JSTACK                                                           SLEVEL12
   10 IF(JSTOP .EQ. 0) GO TO 40                                              SLEVEL13
      NPNTR=BITGET(STACK(JSTOP),60,15)                                       SLEVEL14
      IF(NPNTR .GT. 0 .AND. NPNTR .LT. 77777B) GO TO 20                      SLEVEL15
      IF(BITGET(STACK(JSTOP),30,15) .NE. 0) GO TO 30                         SLEVEL16
      JSTOP=JSTOP-1                                                          SLEVEL17
      GO TO 10                                                               SLEVEL18
   20 JSTOP=NPNTR-1                                                          SLEVEL19
      GO TO 10                                                               SLEVEL20
   30 STACK(JSTACK)=STACK(JSTACK) .AND. 7777777777777700000B                 SLEVEL21
      STACK(JSTACK)=STACK(JSTACK) .OR. JSTOP                                 SLEVEL22
      JAS=BITGET(STACK(JSTOP),30,15)                                         SLEVEL23
      R=BITGET(STACK(JSTOP),15,15)                                           SLEVEL24
      RTEMP=R                                                                SLEVEL25
      LVVPOS =          JAS
      LVVTYP =    3
      LVFUNC=       LEVEL
      LVVARG=           R
      CALL LVFIND(LV2       A,LV2       B,LV2       C,LV2       D)
      LV1     AAD =           R
      IF (LVVAL.NE.-1) LV1     AAD = LVVAL
               R = LV1     AAD
      FAIL=.FALSE.                                                           SLEVEL27
      IF(BITGET(STACK(JSTOP),60,15) .EQ. 77777B)                             SLEVEL28
     $ STACK(JSTOP)=STACK(JSTOP) .AND. 7777777777777700000B                  SLEVEL29
      RETURN                                                                 SLEVEL30
   40 FAIL=.TRUE.                                                            SLEVEL31
      RETURN                                                                 SLEVEL32
      RETURN
25000 CONTINUE
      LV2       A=LV2       B=LV2       C=LV2       D=0
      GO TO 25001
      END
```

GIRL Version

```
$       SUBROUTINE SLEVEL(FAIL)                                          SLEVEL     2
        COMMON /NEED/ START,ASSOC,LEVEL,STOP                             SLEVEL     3
        COMMON/NEEDS/STJ,JSTACK,R,JAS,J,JLAST,RTEMP,STACK(400)           SLEVEL     4
        COMMON /STRING/ NNN(2),STR                                       SLEVEL     5
        COMMON /JL/ JSTOP                                                SLEVEL     6
        INTEGER START,STOP,ASSOC,STACK,STR(1),STJ,R,RTEMP                SLEVEL     7
        INTEGER BITGET                                                   SLEVEL     8
        LOGICAL FAIL                                                     SLEVEL     9
G       EXECUTE                                                          SLEVEL    10
        RTEMP=0                                                          SLEVEL    11
        JSTOP=JSTACK                                                     SLEVEL    12
   10   IF(JSTOP .EQ. 0) GO TO 40                                        SLEVEL    13
        NPNTR=BITGET(STACK(JSTOP),60,15)                                 SLEVEL    14
        IF(NPNTR .GT. 0 .AND. NPNTR .LT. 77777B) GO TO 20                SLEVEL    15
        IF(BITGET(STACK(JSTOP),30,15) .NE. 0) GO TO 30                   SLEVEL    16
        JSTOP=JSTOP-1                                                    SLEVEL    17
        GO TO 10                                                         SLEVEL    18
   20   JSTOP=NPNTR-1                                                    SLEVEL    19
        GO TO 10                                                         SLEVEL    20
   30   STACK(JSTACK)=STACK(JSTACK) .AND. 777777777777777000008          SLEVEL    21
        STACK(JSTACK)=STACK(JSTACK) .OR. JSTOP                           SLEVEL    22
        JAS=BITGET(STACK(JSTOP),30,15)                                   SLEVEL    23
        R=BITGET(STACK(JSTOP),15,15)                                     SLEVEL    24
        RTEMP=R                                                          SLEVEL    25
G       R+LEVEL.JAS 'R                                                   SLEVEL    26
        FAIL=.FALSE.                                                     SLEVEL    27
        IF(BITGET(STACK(JSTOP),60,15) .EQ. 77777B)                       SLEVEL    28
     $  STACK(JSTOP)=STACK(JSTOP) .AND. 777777777777777000008            SLEVEL    29
        RETURN                                                           SLEVEL    30
   40   FAIL=.TRUE.                                                      SLEVEL    31
        RETURN                                                           SLEVEL    32
G       COMPLETE                                                         SLEVEL    33
```

```
      SUBROUTINE SQUEEZ                                                 SQUEEZE    2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH       2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A     80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH      4
      INTEGER A,D,BLANK,AICH                                            SQUEEZE    4
      DATA BLANK/1H /,AICH/1HH/                                         SQUEEZE    5
C** THE PURPOSE OF THIS ROUTINE IS TO SQUEEZE THE BLANKS OUT OF A       SQUEEZE    6
C** CHARACTER STRING "D" WHICH CONSTITUTES A LANGUAGE ELEMENT           SQUEEZE    7
      J=0                                                               SQUEEZE    8
      DO 10 I=1,M                                                       SQUEEZE    9
      IF(D(I) .EQ. BLANK) GO TO 10                                      SQUEEZE   10
      J=J+1                                                             SQUEEZE   11
      D(J)=D(I)                                                         SQUEEZE   12
      IF(D(J) .NE. AICH) GO TO 10                                       SQUEEZE   13
      IF(JTYP .NE. 3) GO TO 10                                          SQUEEZE   14
C** IF CHARACTER STRING CONSTITUTES A HOLLERITH CONSTANT, RETURN        SQUEEZE   15
      M=M+J-I                                                           SQUEEZE   16
      RETURN                                                            SQUEEZE   17
   10 CONTINUE                                                          SQUEEZE   18
      M=J                                                               SQUEEZE   19
C** SET "M" = SIZE OF STRING                                            SQUEEZE   20
      RETURN                                                            SQUEEZE   21
      END                                                               SQUEEZE   22
```

FORTRAN Version

```
      SUBROUTINE SSTOP(FAIL)                                                SSTOP  2
      COMMON/LVARGS/LVFUNC,LVVARG,LVVAD,LVVPOS,LVVTYP,       LVVAL,
     +LVHEAD,LVVNVL,LVDEST,LVVALS(10),LVTYPE(10),LVSKIP
      COMMON/LVTABL/LVTSIZ,LVMAP(     1)/LVVSEQ/LVSIZE,LVSOSP(     1)
      COMMON /NEED/ START,ASSOC,LEVEL,STOP                                  SSTOP  3
      COMMON/NEEDS/STJ,JSTACK,R,JAS,J,JLAST,PTEMP,STACK(400)                SSTOP  4
      COMMON /STRING/ NNN(2),STR                                            SSTOP  5
      INTEGER START,STOP,ASSOC,STACK,STR(1),STJ,R,TEMP                      SSTOP  6
      INTEGER BITGET,BITPUT                                                 SSTOP  7
      LOGICAL FAIL                                                          SSTOP  8
      GO TO 25000
25001 CONTINUE
      JSTOPS=JSTACK                                                         SSTOP 10
    5 CONTINUE
      LVVTYP = 3
      LVVPOS = 1
      LVINDX = 0
      LVFUNC=       ASSOC
      LVVARG=           R
      CALL LVFIND(LVINDX,LVINDX,LVINDX,LVINDX)
      LV1     AAD =           R
      IF (LVVAL.NE.-1) LV1     AAD = LVVAL
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.EQ.-1) GO TO          10
      JSTOPS=JSTOPS+1                                                       SSTOP 12
      ISTCK=BITGET(STACK(JSTOPS),45,15)                                     SSTOP 13
      STACK(JSTOPS)=SHIFT(ISTCK,15) .OR. SHIFT(R,45)                        SSTOP 14
   10 CONTINUE
      LV1     AAD =           R
      LVVAL = -100
      IF (LV1     AAD.NE.       STOP) LVVAL = -1
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.NE.-1) GO TO          20
      LVVTYP = 3
      LVVPOS = 1
      LVINDX = 0
      LVFUNC=        STOP
      LVVARG= LV1     AAD
      CALL LVFIND(LVINDX,LVINDX,LVINDX,LVINDX)
      LV1     AAI = LV1     AAD
      IF (LVVAL.NE.-1) LV1     AAI = LVVAL
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.EQ.-1) GO TO          30
               R = LV1     AAI
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.EQ.-1) GO TO           5
      IF (LVVTR.NE.-1) GO TO           5
   20 FAIL=.FALSE.                                                          SSTOP 16
      RETURN                                                                SSTOP 17
   30 CONTINUE
      LVVAL = -100
      IF (    JSTACK.NE.    JSTOPS) LVVAL = -1
      LVVTR = LVVAL
```

```
      LVVAL = -100
      IF (LVVTR.EQ.-1) GO TO          40
      FAIL=.TRUE.                                                          SSTOP 19
      RETURN                                                               SSTOP 20
   40 R=BITGET(STACK(JSTOPS),15,15)                                        SSTOP 21
      JAS=BITGET(STACK(JSTOPS),30,15)+1                                    SSTOP 22
      LVVPOS =         JAS
      LVVTYP =    3
      LVFUNC=       ASSOC
      LVVARG=           R
      CALL LVFIND(LV2       A,LV2       B,LV2       C,LV2       D)
      LV1     AAD =           R
      IF (LVVAL.NE.-1) LV1     AAD = LVVAL
            TEMP = LV1     AAD
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.NE.-1) GO TO          50
      JSTOPS=JSTOPS-1                                                      SSTOP 24
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.NE.-1) GO TO          30
   50 STACK(JSTOPS)=STACK(JSTOPS) .AND. 777770000077777777778              SSTOP 26
      STACK(JSTOPS)=BITPUT(STACK(JSTOPS),JAS,30)                           SSTOP 27
      LVVAL = -100
      IF (          R.NE.       TEMP) LVVAL = -1
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.NE.-1) GO TO          40
               R =        TEMP
      LVVTR = LVVAL
      LVVAL = -100
      IF (LVVTR.NE.-1) GO TO           5
      RETURN
25000 CONTINUE
      LV2       A=LV2       B=LV2       C=LV2       D=0
      GO TO 25001
      END
```

GIRL Version

```
$     SUBROUTINE SSTOP(FAIL)                                               SSTOP      2
      COMMON /NEED/ START,ASSOC,LEVEL,STOP                                 SSTOP      3
      COMMON/NEEDS/STJ,JSTACK,R,JAS,J,JLAST,RTEMP,STACK(400)               SSTOP      4
      COMMON /STRING/ NNN(2),STR                                           SSTOP      5
      INTEGER START,STOP,ASSOC,STACK,STR(1),STJ,R,TEMP                     SSTOP      6
      INTEGER BITGET,BITPUT                                                SSTOP      7
      LOGICAL FAIL                                                         SSTOP      8
G     EXECUTE                                                              SSTOP      9
      JSTOPS=JSTACK                                                        SSTOP     10
G   5 R+ASSOC/10                                                           SSTOP     11
      JSTOPS=JSTOPS+1                                                      SSTOP     12
      ISTCK=BITGET(STACK(JSTOPS),45,15)                                    SSTOP     13
      STACK(JSTOPS)=SHIFT(ISTCK,15) .OR. SHIFT(R,45)                       SSTOP     14
G  10 R(=STOP//20,+STOP/30 *R/5/5)                                         SSTOP     15
   20 FAIL=.FALSE.                                                         SSTOP     16
      RETURN                                                               SSTOP     17
G  30 JSTACK=JSTOPS/40                                                     SSTOP     18
      FAIL=.TRUE.                                                          SSTOP     19
      RETURN                                                               SSTOP     20
   40 R=BITGET(STACK(JSTOPS),15,15)                                        SSTOP     21
      JAS=BITGET(STACK(JSTOPS),30,15)+1                                    SSTOP     22
G     R+ASSOC.JAS *TEMP//50                                                SSTOP     23
      JSTOPS=JSTOPS-1                                                      SSTOP     24
G     //30                                                                 SSTOP     25
   50 STACK(JSTOPS)=STACK(JSTOPS) .AND. 777770000077777777778              SSTOP     26
      STACK(JSTOPS)=BITPUT(STACK(JSTOPS),JAS,30)                           SSTOP     27
G     R=TEMP//40                                                           SSTOP     28
G     TEMP *R//5                                                           SSTOP     29
G     COMPLETE                                                             SSTOP     30
```

```
      SUBROUTINE STATNO                                                  STATNO     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),   RICH      2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,         CY58A     60
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IOES                                   RICH      4
      COMMON/LABELS/STATRA(2,200),NLABEL                                 STATNO     4
      COMMON/DOLOOP/ISTACK(4,50),NSTACK,ILOOP,IOVFLW                     STATNO     5
      COMMON/BASBLK/IBLOCK(2500),NBLOCK,NB,NBRNCH                        CY58A     26
      INTEGER A,BLANK,STATRA                                             STATNO     7
      INTEGER BITPUT,BITGET                                              STATNO     8
      DATA BLANK/1H /                                                    STATNO     9
C** THIS ROUTINE PROCESSES STATEMENT NUMBERS,BASIC BLOCKS, AND DO LOOPS   STATNO    10
      LOC=0                                                              STATNO    11
      DO 5 I=1,5                                                         STATNO    12
      IF(A(I) .NE. BLANK) GO TO 10                                       STATNO    13
    5 CONTINUE                                                           STATNO    14
C** STATEMENT IS UNLABELLED                                              STATNO    15
      IF(ITYP .NE. 18) GO TO 7                                           STATNO    16
C** END STATEMENT                                                        STATNO    17
      IF(IBLKDT .EQ. 1) RETURN                                           STATNO    18
C** STORE NUMBER OF BRANCHES IN BASIC BLOCK TABLE                        STATNO    19
      IF(NBRNCH .EQ. 0) GO TO 110                                        STATNO    20
      IBLOCK(IBLKST)=BITPUT(IBLOCK(IBLKST),NBRNCH,6)                     STATNO    21
      RETURN                                                             STATNO    22
C** IF FORMAT STATEMENT, ISSUE DIAGNOSTIC                                STATNO    23
    7 IF(ITYP .EQ. 28) GO TO 90                                          STATNO    24
C** UNLABELLED STATEMENT MAY NEED FURTHER PROCESSING                     STATNO    25
      IF(NBLOCK .EQ. 0) GO TO 8                                          STATNO    26
      IF(IBLOCK(NBLOCK) .EQ. 998) GO TO 32                               STATNO    27
      IF(NB .EQ. 2) GO TO 31                                             STATNO    28
      IF(NB .EQ. 1) GO TO 70                                             STATNO    29
      RETURN                                                             STATNO    30
C** FIRST STATEMENT IN PROGRAM - INITIALIZE BASIC BLOCK TABLE            STATNO    31
    8 NBLOCK=1                                                           STATNO    32
      GO TO 34                                                           STATNO    33
C** LABELLED STATEMENT                                                   STATNO    34
   10 IF(ITYP .EQ. 18) GO TO 50                                          STATNO    35
      JPTR=I                                                             STATNO    36
C** GET STATEMENT LABEL AND CHECK THAT IT IS INTEGER                     STATNO    37
      CALL GNLE                                                          STATNO    38
      IF(JTYP .NE. 5) GO TO 50                                           STATNO    39
      IF(A(6) .NE. BLANK) GO TO 50                                       STATNO    40
      IF(JPTR .LT. 6) GO TO 50                                           STATNO    41
C** STORE IN STATEMENT NUMBER TABLE                                      STATNO    42
      CALL STSRCH                                                        STATNO    43
      IF(BITGET(STATRA(2,LOC),9,3) .EQ. 1) GO TO 60                      STATNO    44
C** SET "DEFINED" FLAG                                                   STATNO    45
      STATRA(2,LOC)=BITPUT(STATRA(2,LOC),1,9)                            STATNO    46
      IF(LTYP .EQ. 9) GO TO 20                                           STATNO    47
C** STORE STATEMENT TYPE                                                 STATNO    48
      STATRA(2,LOC)=BITPUT(STATRA(2,LOC),ITYP,6)                         STATNO    49
      GO TO 30                                                           STATNO    50
   20 STATRA(2,LOC)=BITPUT(STATRA(2,LOC),9,6)                            STATNO    51
   30 IF(ITYP .EQ. 28) RETURN                                            STATNO    52
      IF(NBLOCK .EQ. 0) GO TO 8                                          STATNO    53
      IF(NB .EQ. 1) GO TO 32                                             STATNO    54
C** STORE BRANCH FOR PREVIOUS BLOCK                                      STATNO    55
   31 NBLOCK=NBLOCK+1                                                    STATNO    56
```

```
      IBLOCK(NBLOCK)=998                                                    STATNO     57
      NBRNCH=1                                                              STATNO     58
C** CLOSE OUT PREVIOUS BLOCK                                                STATNO     59
   32 NB=0                                                                  STATNO     60
C** INCREMENT BLOCK COUNTER - START OF NEXT BLOCK                           STATNO     61
      NBLOCK=NBLOCK+1                                                       STATNO     62
C** STORE POINTER TO NEXT BLOCK                                             STATNO     63
      IBLOCK(IBLKST)=BITPUT(IBLOCK(IBLKST),NBLOCK,28)                       STATNO     64
C** STORE NO. OF BRANCHES                                                   STATNO     65
      IBLOCK(IBLKST)=BITPUT(IBLOCK(IBLKST),NBRNCH,6)                        STATNO     66
C** CHECK THAT DO INDICES ARE MADE UNDEFINED AT END OF BLOCK                STATNO     67
      J1=IBLKST+1                                                           STATNO     68
      J2=NBLOCK-NBRNCH-1                                                    STATNO     69
      IF(IBLOCK(J2) .GT. 6000) GO TO 34                                     STATNO     70
      J21=J2-1                                                              STATNO     71
   21 IF(IBLOCK(J1) .LT. 6000) GO TO 34                                     STATNO     72
      IRES=IBLOCK(J1)                                                       STATNO     73
      DO 22 K2=J1,J21                                                       STATNO     74
   22 IBLOCK(K2)=IBLOCK(K2+1)                                               STATNO     75
      IBLOCK(J2)=IRES                                                       STATNO     76
      GO TO 21                                                              STATNO     77
C** OPEN THE NEW BLOCK                                                      STATNO     78
   34 IBLKST=NBLOCK                                                         STATNO     79
      NBRNCH=0                                                              STATNO     80
C** STORE DO LOOP POINTER                                                   STATNO     81
      IBLOCK(IBLKST)=BITPUT(0,ILOOP,12)                                     STATNO     82
      IF(LOC .EQ. 0) RETURN                                                 STATNO     83
C** STORE LOCATION OF STATEMENT NUMBER OF BLOCK                             STATNO     84
      IBLOCK(IBLKST)=BITPUT(IBLOCK(IBLKST),LOC,36)                          STATNO     85
C** STORE BLOCK START IN STATEMENT NUMBER TABLE                             STATNO     86
      STATRA(2,LOC)=BITPUT(STATRA(2,LOC),IBLKST,36)                         STATNO     87
      IF(BITGET(STATRA(2,LOC),15,3) .NE. 1) RETURN                          STATNO     88
      IF(IOVFLW .EQ. 1) RETURN                                              STATNO     89
      IF(LOC .NE. ISTACK(1,ILOOP)) GO TO 80                                 STATNO     90
      IF(ITYP .GE. 3 .AND. ITYP .LE. 6) GO TO 100                           STATNO     91
      IF( ITYP .EQ. 9 .OR. ITYP .EQ. 10 .OR. ITYP .EQ. 17) GO TO 100        STATNO     92
C** DO TERMINAL STATEMENT                                                   STATNO     93
      NB=2                                                                  STATNO     94
C** CLOSE OUT LOOP                                                          STATNO     95
      ISTACK(2,ILOOP)=1                                                     STATNO     96
C** STORE DO INDEX IN BASIC BLOCK TABLE                                     STATNO     97
      NBLOCK=NBLOCK+1                                                       STATNO     98
      IBLOCK(NBLOCK)=6000+ISTACK(4,ILOOP)                                   STATNO     99
      KLOOP=ILOOP-1                                                         STATNO    100
C** RESET VALUE OF THE "CURRENT LOOP"                                       STATNO    101
      DO 40 J=1,KLOOP                                                       STATNO    102
      LOOP=ILOOP-J                                                          STATNO    103
      IF(ISTACK(2,LOOP) .EQ. 1) GO TO 40                                    STATNO    104
      IF(ISTACK(1,LOOP) .EQ. LOC) GO TO 35                                  STATNO    105
      ILOOP=LOOP                                                            STATNO    106
      RETURN                                                                STATNO    107
   35 ISTACK(2,LOOP)=1                                                      STATNO    108
      NBLOCK=NBLOCK+1                                                       STATNO    109
      IBLOCK(NBLOCK)=6000+ISTACK(4,LOOP)                                    STATNO    110
   40 CONTINUE                                                              STATNO    111
      ILOOP=0                                                               STATNO    112
      RETURN                                                                STATNO    113
```

```
   50 IERC=32                                                          STATNO   114
      GO TO 200                                                        STATNO   115
   60 IERC=33                                                          STATNO   116
      GO TO 200                                                        STATNO   117
   70 IERC=34                                                          STATNO   118
      GO TO 200                                                        STATNO   119
   80 IERC=35                                                          STATNO   120
      GO TO 200                                                        STATNO   121
   90 IERC=36                                                          STATNO   122
      GO TO 200                                                        STATNO   123
  100 IERC=37                                                          STATNO   124
      GO TO 200                                                        STATNO   125
  110 IERC=38                                                          STATNO   126
  200 CALL ERROR(IERC)                                                 STATNO   127
      RETURN                                                           STATNO   128
      END                                                              STATNO   129
```

```
      SUBROUTINE STFNC(NFNC)                                           CY58A     47
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH      2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A     80
     2 LTYP,ITYP,I8LKOT,MODE,IERR,IOES                                 RICH      4
      COMMON/FUNC/IFNCRA(5,12),MARGS,IARGS(50),FNCLOC(5),NFUNC         CY58A     48
      INTEGER BITGET                                                   STFUNC    5
C** STATEMENT FUNCTION PROCESSOR                                       STFUNC    6
C** GET NO. OF ARGUMENTS FROM SYMBOL TABLE                             STFUNC    7
      NARG=BITGET(IDTBL(3,LOC),7,6)                                    STFUNC    8
C** CHECK CORRECTNESS OF NO. OF ARGUMENTS                              STFUNC    9
      NAR2=IFNCRA(NFNC,1)                                              STFUNC   10
      IF(NARG .NE. NAR2) CALL ERROR(26,IDTBL(1,LOC))                   STFUNC   11
      NARGS=MIN0(NARG,NAR2)                                            STFUNC   12
      KOUNT=0                                                          STFUNC   13
      NT=1+(NARGS-1)/6                                                 STFUNC   14
C** THESE TWO LOOPS CHECK TYPE AND DIMENSIONALITY OF ARGUMENTS         STFUNC   15
      DO 10 I=1,NT                                                     STFUNC   16
      ICOL1=-6                                                         STFUNC   17
      ICOL2=-3                                                         STFUNC   18
      DO 10 J=1,6                                                      STFUNC   19
      KOUNT=KOUNT+1                                                    STFUNC   20
      IF(KOUNT .GT. NARGS) RETURN                                      STFUNC   21
      ICOL1=ICOL1+9                                                    STFUNC   22
      ICOL2=ICOL2+9                                                    STFUNC   23
C** GET DIMENSIONALITY - IF NOT 0, ISSUE DIAGNOSTIC                    STFUNC   24
      IF(BITGET(IFNCRA(NFNC,I+1),ICOL2,3) .NE. 0) CALL ERROR(50,KOUNT) STFUNC   25
C** GET TYPE                                                           STFUNC   26
      ITP1=BITGET(IFNCRA(NFNC,I+1),ICOL1,3)                            STFUNC   27
      IF(ITP1 .EQ. 0) GO TO 10                                         STFUNC   28
C** CHECK CORRECTNESS OF TYPE                                          STFUNC   29
      ITP2=BITGET(IDTBL(3,LOC+KOUNT),10,3)                             STFUNC   30
      IF(ITP1 .NE. ITP2) CALL ERROR(51,KOUNT)                          STFUNC   31
   10 CONTINUE                                                         STFUNC   32
      RETURN                                                           STFUNC   33
      END                                                              STFUNC   34
```

```
      SUBROUTINE STORE                                                  STORE     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH      2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A    80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH      4
C** THIS ROUTINE STORES A NAME "NXTID" IN THE SYMBOL TABLE AND UPDATES   STORE     4
C** SYMBOL TABLE POINTERS                                               STORE     5
      NID=NID+1                                                         STORE     6
      IF(NID .GT. 500) GO TO 20                                         STORE     7
      IF(INITID(IDTYP) .NE. 0) GO TO 5                                  STORE     8
      INITID(IDTYP)=NID                                                 STORE     9
    5 CONTINUE                                                          STORE    10
      IDTBL(1,NID)=NXTID                                                STORE    11
      IDTBL(2,NID)=0                                                    STORE    12
      IF(LASTID(IDTYP) .EQ. 0) GO TO 10                                 STORE    13
      LAST=LASTID(IDTYP)                                                STORE    14
      IDTBL(2,LAST)=NID                                                 STORE    15
   10 LASTID(IDTYP)=NID                                                 STORE    16
      RETURN                                                            STORE    17
   20 WRITE(6,25)                                                       STORE    18
   25 FORMAT(/////5X,46H SYMBOL TABLE OVERFLOW - PROCESSING TERMINATED)  STORE    19
      STOP                                                              STORE    20
      END                                                               STORE    21
```

```
      SUBROUTINE STSRCH                                                 STSRCH    2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH      2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A    80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH      4
      COMMON/LABELS/STATRA(2,200),NLABEL                                STSRCH    4
      INTEGER STATRA                                                    STSRCH    5
C** THIS ROUTINE SEARCHES THE STATEMENT NUMBER TABLE AND STORES THE      STSRCH    6
C** NUMBER IN THE TABLE IF NOT FOUND                                    STSRCH    7
      IF(NLABEL .EQ. 0) GO TO 15                                        STSRCH    8
      DO 10 I=1,NLABEL                                                  STSRCH    9
      IF(STATRA(1,I) .NE. N2) GO TO 10                                  STSRCH   10
C** STATEMENT NUMBER FOUND IN TABLE - RETURN LOCATION WHERE FOUND        STSRCH   11
      LOC=I                                                             STSRCH   12
      RETURN                                                            STSRCH   13
   10 CONTINUE                                                          STSRCH   14
C** STATEMENT NUMBER NOT FOUND - INCREMENT COUNTER                      STSRCH   15
   15 NLABEL=NLABEL+1                                                   STSRCH   16
      IF(NLABEL .GT. 200) GO TO 20                                      STSRCH   17
C** STORE STATEMENT NUMBER AND RETURN LOCATION WHERE STORED             STSRCH   18
      LOC=NLABEL                                                        STSRCH   19
      STATRA(1,LOC)=N2                                                  STSRCH   20
      RETURN                                                            STSRCH   21
   20 WRITE(6,25)                                                       STSRCH   22
   25 FORMAT(/////5X,53H STATEMENT NO. TABLE OVERFLOW - PROCESSING TERMI STSRCH   23
     *NATED)                                                            STSRCH   24
      STOP                                                              STSRCH   25
      END                                                               STSRCH   26
```

```
      SUBROUTINE SUB                                                    SUB      2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH     2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A    80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IOES                                  RICH     4
      DIMENSION IALPH(10),IALPH2(8),KT(5)                               SUB      4
      INTEGER BLANK,COMMA,RPAR,A,D                                      SUB      5
      INTEGER BITPUT                                                    SUB      6
      DATA (IALPH(I),I=1,10)/1HS,1HU,1HB,1HR,1HO,1HU,1HT,1HI,1HN,1HE/   SUB      7
      DATA (IALPH2(I),I=1,8)/1HF,1HU,1HN,1HC,1HT,1HI,1HO,1HN/           SUB      8
      DATA (KT(I),I=1,5)/1HR,1HC,1HD,1HI,1HL/                           SUB      9
      DATA BLANK/1H /,LPAR/1H(/,RPAR/1H)/,COMMA/1H,/                    SUB     10
C** SUBROUTINE AND FUNCTION STATEMENT PROCESSOR                         SUB     11
      NARG=0                                                            SUB     12
      ISTATE=0                                                          SUB     13
      IF(ITYP .EQ. 30) GO TO 5                                          SUB     14
C** FUNCTION STATEMENT                                                  SUB     15
    2 DO 3 I=1,8                                                        SUB     16
      IF(NEXT(JPTR) .NE. IALPH2(I)) GO TO 50                            SUB     17
    3 CONTINUE                                                          SUB     18
      GO TO 12                                                          SUB     19
C** SUBROUTINE STATEMENT                                                SUB     20
    5 DO 10 I=1,10                                                      SUB     21
      IF(NEXT(JPTR) .NE. IALPH(I)) GO TO 50                             SUB     22
   10 CONTINUE                                                          SUB     23
      GO TO 17                                                          SUB     24
   12 IPTR=JPTR                                                         SUB     25
      IFRST=NEXT(1)                                                     SUB     26
      IF(IFRST .EQ. 1HF) GO TO 14                                       SUB     27
C** FUNCTION IS A TYPED FUNCTION, GET TYPE                              SUB     28
      DO 13 I=1,5                                                       SUB     29
      IF(IFRST .NE. KT(I)) GO TO 13                                     SUB     30
      ISTATE=I                                                          SUB     31
      GO TO 14                                                          SUB     32
   13 CONTINUE                                                          SUB     33
   14 JPTR=IPTR                                                         SUB     34
C** GET NAME OF SUBROUTINE OR FUNCTION AND STORE IN SYMBOL TABLE        SUB     35
   17 CALL GNLE                                                         SUB     36
      IF(JTYP .NE. 2) GO TO 50                                          SUB     37
      IDTYP=2                                                           SUB     38
      CALL STORE                                                        SUB     39
      IE=D(1)                                                           SUB     40
      IF(ITYP .NE. 31) GO TO 15                                         SUB     41
C** FUNCTION MUST BE FOLLOWED BY ARGUMENT LIST                          SUB     42
      IF(NEXT(JPTR) .NE. LPAR) GO TO 50                                 SUB     43
      IFNCNM=NXTID                                                      SUB     44
      GO TO 20                                                          SUB     45
   15 IF(NEXT(JPTR) .EQ. BLANK) GO TO 30                                SUB     46
      IF(A(JPTR-1) .NE. LPAR) GO TO 50                                  SUB     47
C** GET NEXT ARGUMENT                                                   SUB     48
   20 CALL GNLE                                                         SUB     49
      IF(JTYP .NE. 2) GO TO 60                                          SUB     50
      CALL SEARCH                                                       SUB     51
      IF(ISRCH(1) .NE. 0 .OR.  ISRCH(2) .NE. 0) CALL ERROR(86,NXTID)    SUB     52
      IDTYP=1                                                           SUB     53
      CALL STORE                                                        SUB     54
C** SET "FORMAL PARAMETER" FLAG                                         SUB     55
      IDTBL(3,NID)=BITPUT(IDTBL(3,NID),1,12)                            SUB     56
```

```
C** INCREMENT ARGUMENT COUNTER                                    SUB        57
      NARG=NARG+1                                                 SUB        58
      IF(NEXT(JPTR) .EQ. RPAR) GO TO 30                           SUB        59
      IF(A(JPTR-1) .NE. COMMA) GO TO 50                           SUB        60
      GO TO 20                                                    SUB        61
   30 LOC=1                                                       SUB        62
      D(1)=IE                                                     SUB        63
C** CHECK NO. OF ARGUMENTS AND STORE IN SYMBOL TABLE              SUB        64
      IF(NARG .GT. 63) CALL ERROR(83)                             SUB        65
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),NARG,7)                    SUB        66
      IF(ITYP .EQ. 30) RETURN                                     SUB        67
      IF(ISTATE .EQ. 0) GO TO 55                                  SUB        68
C** IF TYPED FUNCTION, STORE TYPE AND SET "TYPE SET" FLAG         SUB        69
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),ISTATE,10)                 SUB        70
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),1,11)                      SUB        71
      RETURN                                                      SUB        72
C** SET FUNCTION TYPE IMPLICITLY                                  SUB        73
   55 CALL IMPTYP                                                 SUB        74
      RETURN                                                      SUB        75
   50 CALL ERROR(7)                                               SUB        76
      RETURN                                                      SUB        77
   60 CALL ERROR(86,NXTID)                                        SUB        78
      RETURN                                                      SUB        79
      END                                                         SUB        80
```

```
      SUBROUTINE SUBCHK                                                 SUBCHK     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH       2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A     80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH       4
      COMMON/GLOBAL/NBLK,NREF,NSUBS,BLKTBL(200),EXTTBL(100),ISUBS(100)  SUBCHK     4
      COMMON/LIST/NLIST,NINTFC,ISUBLT(2,200),INTFAC(300)                SUBCHK     5
      INTEGER BITPUT,BITGET                                             SUBCHK     6
C** AFTER PROCESSING OF THE MODULE IS COMPLETED, THIS ROUTINE IS CALLED  SUBCHK     7
C** TO PROCESS THE SUBROUTINE NAME AND ARGUMENT LIST                     SUBCHK     8
      IF(IBLKDT .EQ. 1) RETURN                                          SUBCHK     9
C** INCREMENT SUBROUTINE COUNTER                                        SUBCHK    10
      NSUBS=NSUBS+1                                                     SUBCHK    11
      IF(NSUBS .GT. 100) GO TO 50                                       SUBCHK    12
C** GET NUMBER OF ARGUMENTS AND TYPE                                    SUBCHK    13
      NARG=BITGET(IDTBL(3,1),7,6)                                       SUBCHK    14
      ITP=BITGET(IDTBL(3,1),10,3)                                       SUBCHK    15
      IF(BITGET(IDTBL(3,1),16,1) .EQ. 1) GO TO 15                       SUBCHK    16
C** MODULE ARGUMENT LIST HAS NOT YET BEEN ENCOUNTERED - SET FLAG        SUBCHK    17
      IDTBL(3,1)=BITPUT(IDTBL(3,1),1,16)                                SUBCHK    18
C** SEARCH SESCOMP LIST                                                 SUBCHK    19
      DO 5 I=1,NLIST                                                    SUBCHK    20
      IF(IDTBL(1,1) .NE. ISUBLT(1,I)) GO TO 5                           SUBCHK    21
C** NAME FOUND IN SESCOMP LIST                                          SUBCHK    22
      LISTLC=I                                                          SUBCHK    23
C** STORE LIST LOCATION IN SYMBOL TABLE                                 SUBCHK    24
      IDTBL(3,1)=BITPUT(IDTBL(3,1),LISTLC,36)                           SUBCHK    25
      GO TO 20                                                          SUBCHK    26
    5 CONTINUE                                                          SUBCHK    27
C** NAME NOT FOUND IN SESCOMP LIST - ISSUE DIAGNOSTIC                   SUBCHK    28
      CALL ERROR(52)                                                    SUBCHK    29
C** ADD NAME TO SESCOMP LIST                                            SUBCHK    30
      NLIST=NLIST+1                                                     SUBCHK    31
      ISUBLT(1,NLIST)=IDTBL(1,1)                                        SUBCHK    32
C** ADD NAME TO SUBROUTINE TABLE                                        SUBCHK    33
      ISUBS(NSUBS)=NLIST                                                SUBCHK    34
      ISUBLT(2,NLIST)=0                                                 SUBCHK    35
C** STORE LIST LOCATION IN SYMBOL TABLE                                 SUBCHK    36
      IDTBL(3,1)=BITPUT(IDTBL(3,1),NLIST,36)                            SUBCHK    37
      IF(NARG .EQ. 0) RETURN                                            SUBCHK    38
C** MODULE HAS ARGUMENTS-STORE ATTRIBUTES IN INTERFACE DEFINITION TABLE  SUBCHK    39
      IPTR=NINTFC+1                                                     SUBCHK    40
C** STORE POINTER TO INTERFACE DEFINITION TABLE AND NUMBER OF ARGS.      SUBCHK    41
      ISUBLT(2,NLIST)=BITPUT(IPTR,NARG,6)                               SUBCHK    42
C** STORE TYPE                                                          SUBCHK    43
      ISUBLT(2,NLIST)=BITPUT(ISUBLT(2,NLIST),ITP,13)                    SUBCHK    44
      NINTFC=IPTR+(NARG-1)/6                                            SUBCHK    45
      KOUNT=0                                                           SUBCHK    46
C** THESE TWO LOOPS CONSTRUCT THE INTERFACE DEFINITION FOR THE MODULE    SUBCHK    47
      DO 10 I=IPTR,NINTFC                                               SUBCHK    48
      INTFAC(I)=0                                                       SUBCHK    49
      ICOL1=-6                                                          SUBCHK    50
      ICOL2=-3                                                          SUBCHK    51
      DO 10 J=1,6                                                       SUBCHK    52
      KOUNT=KOUNT+1                                                     SUBCHK    53
      IF(KOUNT .GT. NARG) RETURN                                        SUBCHK    54
      ICOL1=ICOL1+9                                                     SUBCHK    55
      ICOL2=ICOL2+9                                                     SUBCHK    56
```

```
C** GET TYPE AND DIMENSIONALITY OF NEXT ARGUMENT                          SUBCHK     57
      ITP=BITGET(IDTBL(3,KOUNT+1),10,3)                                   SUBCHK     58
      NDIM=BITGET(IDTBL(3,KOUNT+1),7,6)                                   SUBCHK     59
      IDTBL(3,KOUNT+1)=BITPUT(IDTBL(3,KOUNT+1),1,37)                      SUBCHK     60
C** STORE IN INTERFACE DEFINITION                                         SUBCHK     61
      INTFAC(I)=BITPUT(INTFAC(I),ITP,ICOL1)                               SUBCHK     62
   10 INTFAC(I)=BITPUT(INTFAC(I),NDIM,ICOL2)                              SUBCHK     63
      RETURN                                                              SUBCHK     64
C** MODULE PREVIOUSLY ENCOUNTERED - GET SESCOMP LIST LOCATION             SUBCHK     65
C** FROM SYMBOL TABLE                                                    SUBCHK     66
   15 LISTLC=BITGET(IDTBL(3,1),36,9)                                      SUBCHK     67
C** STORE NAME IN SUBROUTINE TABLE                                        SUBCHK     68
   20 ISUBS(NSUBS)=LISTLC                                                 SUBCHK     69
C** GET MODULE CLASS AND NO. OF ARGUMENTS                                 SUBCHK     70
      KLAS=BITGET(ISUBLT(2,LISTLC),10,4)                                  SUBCHK     71
      NAR2=BITGET(ISUBLT(2,LISTLC),6,6)                                   SUBCHK     72
C** CHECK NO. OF ARGUMENTS                                               SUBCHK     73
      IF(NARG .NE. NAR2) CALL ERROR(26,IDTBL(1,1))                        SUBCHK     74
      NARGS=MIN0(NARG,NAR2)                                               SUBCHK     75
C** CHECK TYPE                                                           SUBCHK     76
      IF(ITP.NE.BITGET(ISUBLT(2,LISTLC),13,3))CALL ERROR(49,IDTBL(1,1))   SUBCHK     77
      IF(NARGS .EQ. 0) RETURN                                             SUBCHK     78
C** MODULE HAS ARGUMENTS - CHECK INTERFACE DEFINITION FOR VALIDITY        SUBCHK     79
C** COMPUTE INTERFACE DEFINITION TABLE POINTERS                           SUBCHK     80
      IPTR=BITGET(ISUBLT(2,LISTLC),60,15)                                 SUBCHK     81
      NDPTR=IPTR+(NARGS-1)/6                                              SUBCHK     82
      KOUNT=0                                                             SUBCHK     83
C** THESE TWO LOOPS CHECK THE ARGUMENTS AGAINST THE INTERFACE DEFINITION  SUBCHK     84
      DO 25 I=IPTR,NDPTR                                                  SUBCHK     85
      ICOL1=-6                                                            SUBCHK     86
      ICOL2=-3                                                            SUBCHK     87
      DO 25 J=1,6                                                         SUBCHK     88
      KOUNT=KOUNT+1                                                       SUBCHK     89
      IF(KOUNT .GT. NARGS) RETURN                                         SUBCHK     90
      ICOL1=ICOL1+9                                                       SUBCHK     91
      ICOL2=ICOL2+9                                                       SUBCHK     92
C** GET TYPE AND DIMENSIONALITY FROM INTERFACE DEFINITION TABLE           SUBCHK     93
      ITP=BITGET(INTFAC(I),ICOL1,3)                                       SUBCHK     94
      NDIM=BITGET(INTFAC(I),ICOL2,3)                                      SUBCHK     95
C** GET TYPE AND DIMENSIONALITY FROM SYMBOL TABLE                         SUBCHK     96
      ITP2=BITGET(IDTBL(3,KOUNT+1),10,3)                                  SUBCHK     97
      NDIM2=BITGET(IDTBL(3,KOUNT+1),7,6)                                  SUBCHK     98
C** GET I/O STATUS FROM INTERFACE DEFINITION TABLE                        SUBCHK     99
      IOSTAT=BITGET(INTFAC(I),ICOL2+3,3)                                  SUBCHK    100
      IF(IOSTAT .EQ. 2 .OR. KLAS .EQ. 0) IOSTAT=1                         SUBCHK    101
      IDTBL(3,KOUNT+1)=BITPUT(IDTBL(3,KOUNT+1),IOSTAT,37)                 SUBCHK    102
C** CHECK DIMENSIONALITY                                                 SUBCHK    103
      IF(NDIM .NE. NDIM2) CALL ERROR(50,KOUNT)                            SUBCHK    104
      IF(ITP2 .NE. 0) GO TO 23                                            SUBCHK    105
      ITP2=1                                                              SUBCHK    106
      IFST=BITGET(IDTBL(1,KOUNT+1),6,6)                                   SUBCHK    107
      IF(IFST .LE. 14 .AND. IFST .GE. 9) ITP2=4                           SUBCHK    108
C** CHECK TYPE                                                           SUBCHK    109
   23 IF(ITP .NE. ITP2) CALL ERROR(51,KOUNT)                              SUBCHK    110
   25 CONTINUE                                                            SUBCHK    111
      RETURN                                                              SUBCHK    112
   50 CALL ERROR(25)                                                      SUBCHK    113
      STOP                                                                SUBCHK    114
      END                                                                 SUBCHK    115
```

```
      SUBROUTINE SWITCH                                                 SWITCH     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH       2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A     80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH       4
C** THIS ROUTINE SWITCHES THE SYMBOL TABLE STORAGE OF A NAME FROM        SWITCH     4
C** VARIABLE TO FUNCTION                                                SWITCH     5
      DO 20 I=1,NID                                                     SWITCH     6
      IF(IDTBL(2,I) .NE. LOC) GO TO 20                                  SWITCH     7
      IDTBL(2,I)=IDTBL(2,LOC)                                           SWITCH     8
C** CHANGE SYMBOL TABLE POINTER                                         SWITCH     9
      IF(LASTID(1) .EQ. LOC) LASTID(1)=I                                SWITCH    10
      GO TO 30                                                          SWITCH    11
   20 CONTINUE                                                          SWITCH    12
      INITID(1)=IDTBL(2,LOC)                                            SWITCH    13
C** ADD NAME TO FUNCTION LIST                                           SWITCH    14
   30 LAST=LASTID(2)                                                    SWITCH    15
      IDTBL(2,LAST)=LOC                                                 SWITCH    16
      IDTBL(2,LOC)=0                                                    SWITCH    17
      LASTID(2)=LOC                                                     SWITCH    18
      CALL ERROR(87,IDTBL(1,LOC))                                       SWITCH    19
      RETURN                                                            SWITCH    20
      END                                                               SWITCH    21
```

```
      SUBROUTINE SYMTAB                                                 SYMTAB     2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),  RICH       2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,        CY58A     80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                  RICH       4
      COMMON/LABELS/STATRA(2,200),NLABEL                                SYMTAB     4
      COMMON/LIST/NLIST,NINTFC,ISUBLT(2,200),INTFAC(300)                SYMTAB     5
      COMMON/STFUNC/NSTFNC,ISTFNC(10)                                   SYMTAB     6
      DIMENSION ITABEL(5)                                               SYMTAB     7
      INTEGER TYPE(5),DIMS(3),BITGET,STATRA                             SYMTAB     8
      DATA (TYPE(I),I=1,5)/4HREAL,6HCOMPLX,6HDOUBLE,6HINTEGR,6HLOGICL/  SYMTAB     9
      DATA (DIMS(I),I=1,3)/1H1,1H2,1H3/                                 SYMTAB    10
C** THIS ROUTINE DISPLAYS THE SYMBOL TABLE                              SYMTAB    11
      IF(NID .LE. 1) RETURN                                             SYMTAB    12
      IF(IBLKDT .EQ. 1) GO TO 2                                         SYMTAB    13
      INTL=INITID(2)                                                    SYMTAB    14
C** DISPLAY HEADING                                                     SYMTAB    15
      WRITE(6,1) IDTBL(1,INTL)                                          SYMTAB    16
    1 FORMAT(//////45X,25H SYMBOL TABLE FOR MODULE ,A6)                 SYMTAB    17
      GO TO 4                                                           SYMTAB    18
    2 WRITE(6,3)                                                        SYMTAB    19
    3 FORMAT(//////46X,28H SYMBOL TABLE FOR BLOCK DATA)                 SYMTAB    20
    4 LOC=INITID(1)                                                     SYMTAB    21
      IF(LOC .EQ. 0) GO TO 28                                           SYMTAB    22
C** DISPLAY VARIABLES                                                   SYMTAB    23
      WRITE(6,5)                                                        SYMTAB    24
    5 FORMAT(//56X,9HVARIABLES/30X,4HNAME,12X,4HTYPE,31X,10HRELOCATION) SYMTAB    25
  100 ITABEL(1)=IDTBL(1,LOC)                                            SYMTAB    26
C** SKIP IF NOT USED                                                    SYMTAB    27
      IF(BITGET(IDTBL(3,LOC),11,1) .EQ. 0) GO TO 27                     SYMTAB    28
C** GET TYPE                                                            SYMTAB    29
      I=BITGET(IDTBL(3,LOC),10,3)                                       SYMTAB    30
      ITABEL(2)=TYPE(I)                                                 SYMTAB    31
      IF(BITGET(IDTBL(3,LOC),1,1) .EQ. 0) GO TO 16                      SYMTAB    32
C** GET DIMENSIONALITY                                                  SYMTAB    33
      ITABEL(3)=5HARRAY                                                 SYMTAB    34
      I=BITGET(IDTBL(3,LOC),7,6)                                        SYMTAB    35
      ITABEL(4)=DIMS(I)                                                 SYMTAB    36
      GO TO 18                                                          SYMTAB    37
   16 ITABEL(3)=1H                                                      SYMTAB    38
      ITABEL(4)=1H                                                      SYMTAB    39
   18 IF(BITGET(IDTBL(3,LOC),12,1) .EQ. 0) GO TO 20                     SYMTAB    40
C** VARIABLE IS FORMAL PARAMETER                                        SYMTAB    41
      ITABEL(5)=5HF. P.                                                 SYMTAB    42
      GO TO 25                                                          SYMTAB    43
   20 IF(BITGET(IDTBL(3,LOC),16,1) .EQ. 1) GO TO 22                     SYMTAB    44
      ITABEL(5)=1H                                                      SYMTAB    45
      GO TO 25                                                          SYMTAB    46
C** VARIABLE IN COMMON - GET COMMON BLOCK NAME                          SYMTAB    47
   22 ICOMNM=IDTBL(6,LOC)                                               SYMTAB    48
      ITABEL(5)=IDTBL(1,ICOMNM)                                         SYMTAB    49
      IF(ITABEL(5) .EQ. 1H ) ITABEL(5)=2H//                             SYMTAB    50
C** DISPLAY LINE                                                        SYMTAB    51
   25 WRITE(6,26) (ITABEL(I),I=1,5)                                     SYMTAB    52
   26 FORMAT(30X,A6,11X,A6,14X,A5,1X,A1,7X,A6)                          SYMTAB    53
   27 LOC=IDTBL(2,LOC)                                                  SYMTAB    54
      IF(LOC .NE. 0) GO TO 100                                          SYMTAB    55
   28 IF(IBLKDT .EQ. 1) GO TO 60                                        SYMTAB    56
```

```
      LOC=IDTBL(2,INTL)                                                  SYMTAB     57
      IF(LOC .EQ. 0) GO TO 60                                            SYMTAB     58
C** DISPLAY EXTERNALS                                                    SYMTAB     59
      WRITE(6,31)                                                        SYMTAB     60
   31 FORMAT(//55X,10H EXTERNALS/44X,4HNAME,10X,4HTYPE,10X,4HARGS)       SYMTAB     61
   30 ITABEL(1)=IDTBL(1,LOC)                                             SYMTAB     62
C** GET SESCOMP LIST LOCATION                                            SYMTAB     63
      LISTLC=BITGET(IDTBL(3,LOC),36,9)                                   SYMTAB     64
      IF(LISTLC .EQ. 0) GO TO 39                                         SYMTAB     65
C** GET TYPE                                                             SYMTAB     66
      ITP=BITGET(ISUBLT(2,LISTLC),13,3)                                  SYMTAB     67
      IF(ITP .EQ. 0) GO TO 32                                            SYMTAB     68
      ITABEL(2)=TYPE(ITP)                                                SYMTAB     69
      GO TO 35                                                           SYMTAB     70
   32 ITABEL(2)=1H                                                       SYMTAB     71
   35 IF(BITGET(ISUBLT(2,LISTLC),14,1) .EQ. 1) GO TO 37                  SYMTAB     72
C** GET NUMBER OF ARGUMENTS                                              SYMTAB     73
      ITABEL(3)=BITGET(ISUBLT(2,LISTLC),6,6)                             SYMTAB     74
C** DISPLAY LINE                                                         SYMTAB     75
      WRITE(6,36) (ITABEL(I),I=1,3)                                      SYMTAB     76
   36 FORMAT(44X,A6,8X,A6,8X,I2)                                         SYMTAB     77
      GO TO 39                                                           SYMTAB     78
   37 WRITE(6,38) ITABEL(1),ITABEL(2)                                    SYMTAB     79
   38 FORMAT(44X,A6,8X,A6,8X,2H>1)                                       SYMTAB     80
   39 LOC=IDTBL(2,LOC)                                                   SYMTAB     81
      IF(LOC .NE. 0) GO TO 30                                            SYMTAB     82
   60 IF(NSTFNC .EQ. 0) GO TO 40                                         SYMTAB     83
C** DISPLAY STATEMENT FUNCTIONS                                          SYMTAB     84
      WRITE(6,62)                                                        SYMTAB     85
   62 FORMAT(//50X,20H STATEMENT FUNCTIONS/                              SYMTAB     86
     $ 44X,4HNAME,10X,4HTYPE,10X,4HARGS)                                 SYMTAB     87
      DO 70 I=1,NSTFNC                                                   SYMTAB     88
C** GET SYMBOL TABLE LOCATION                                            SYMTAB     89
      LC=ISTFNC(I)                                                       SYMTAB     90
C** GET TYPE                                                             SYMTAB     91
      ITP=BITGET(IDTBL(3,LC),10,3)                                       SYMTAB     92
C** GET NUMBER OF ARGUMENTS                                              SYMTAB     93
      NRG=BITGET(IDTBL(3,LC),7,6)                                        SYMTAB     94
C** DISPLAY LINE                                                         SYMTAB     95
   70 WRITE(6,36) IDTBL(1,LC),TYPE(ITP),NRG                              SYMTAB     96
   40 IF(NLABEL .EQ. 0) GO TO 50                                         SYMTAB     97
C** DISPLAY STATEMENT NUMBERS                                            SYMTAB     98
      WRITE(6,42)                                                        SYMTAB     99
   42 FORMAT(//51X,17H STATEMENT LABELS)                                 SYMTAB    100
      WRITE(6,45) (STATRA(1,I),I=1,NLABEL)                               SYMTAB    101
   45 FORMAT(40X,5I8)                                                    SYMTAB    102
      DO 47 I=1,NLABEL                                                   SYMTAB    103
      IF(BITGET(STATRA(2,I),9,3) .NE. 1) CALL ERROR(15,STATRA(1,I))      SYMTAB    104
      IF(BITGET(STATRA(2,I),12,3) .NE. 1) CALL ERROR(16,STATRA(1,I))     SYMTAB    105
   47 CONTINUE                                                           SYMTAB    106
   50 LOC=INITID(3)                                                      SYMTAB    107
      IF(LOC .EQ. 0) RETURN                                              SYMTAB    108
C** DISPLAY COMMON BLOCKS                                                SYMTAB    109
      WRITE(6,52)                                                        SYMTAB    110
   52 FORMAT(//53X,14H COMMON BLOCKS/50X,4HNAME,10X,6HLENGTH)            SYMTAB    111
   51 ITABEL(1)=IDTBL(1,LOC)                                             SYMTAB    112
      IF(ITABEL(1) .EQ. 1H ) ITABEL(1)=2H//                              SYMTAB    113
      WRITE(6,55) ITABEL(1),IDTBL(4,LOC)                                 SYMTAB    114
   55 FORMAT(50X,A6,8X,I8)                                               SYMTAB    115
      LOC=IDTBL(2,LOC)                                                   SYMTAB    116
      IF(LOC .NE. 0) GO TO 51                                            SYMTAB    117
      RETURN                                                             SYMTAB    118
      END                                                                SYMTAB    119
```

```
      SUBROUTINE TYPE                                                    TYPE      2
      COMMON A(1326),D(500),IDTBL(8,500),INITID(3),LASTID(3),ISRCH(3),   RICH      2
     * JPTR,N,M,JTYP,LSTART,N2,IFNCNM,LOGID,NXTID,IDTYP,NID,LOC,         CY58A    80
     2 LTYP,ITYP,IBLKDT,MODE,IERR,IDES                                   RICH      4
      DIMENSION IALPH1(7),IALPH2(7),IALPH3(4),IALPH4(7),IALPH5(15),      TYPE      4
     1 IDIM(3)                                                           TYPE      5
      INTEGER A,RPAR,COMMA,BLANK                                         TYPE      6
      INTEGER BITPUT,BITGET,COMLOC                                       TYPE      7
      DATA (IALPH1(I),I=1,7)/1HL,1HO,1HG,1HI,1HC,1HA,1HL/,               TYPE      8
     1 (IALPH2(I),I=1,7) /1HI,1HN,1HT,1HE,1HG,1HE,1HR/,                 TYPE      9
     2 (IALPH3(I),I=1,4)/1HR,1HE,1HA,1HL/,                              TYPE     10
     3 (IALPH4(I),I=1,7)/1HC,1HO,1HM,1HP,1HL,1HE,1HX/,                  TYPE     11
     4 (IALPH5(I),I=1,15)/1HD,1HO,1HU,1HB,1HL,1HE,1HP,1HR,1HE,1HC,      TYPE     12
     5 1HI,1HS,1HI,1HO,1HN/                                             TYPE     13
      DATA LPAR/1H(/,RPAR/1H)/,COMMA/1H,/,BLANK/1H /                     TYPE     14
C** TYPE STATEMENT PROCESSOR                                             TYPE     15
      MUL=1                                                              TYPE     16
      IT=ITYP-18                                                         TYPE     17
      GO TO (10,20,30,40,50),IT                                          TYPE     18
C** INTEGER STATEMENT                                                    TYPE     19
   10 DO 15 I=1,7                                                        TYPE     20
      IF(NEXT(JPTR) .NE. IALPH2(I)) GO TO 110                            TYPE     21
   15 CONTINUE                                                           TYPE     22
      ISTATE=4                                                           TYPE     23
      GO TO 60                                                           TYPE     24
C** REAL STATEMENT                                                       TYPE     25
   20 DO 25 I=1,4                                                        TYPE     26
      IF(NEXT(JPTR) .NE. IALPH3(I)) GO TO 110                            TYPE     27
   25 CONTINUE                                                           TYPE     28
      ISTATE=1                                                           TYPE     29
      GO TO 60                                                           TYPE     30
C** DOUBLE PRECISION STATEMENT                                           TYPE     31
   30 DO 35 I=1,15                                                       TYPE     32
      IF(NEXT(JPTR) .NE. IALPH5(I)) GO TO 110                            TYPE     33
   35 CONTINUE                                                           TYPE     34
      MUL=2                                                              TYPE     35
      ISTATE=3                                                           TYPE     36
      GO TO 60                                                           TYPE     37
C** COMPLEX STATEMENT                                                    TYPE     38
   40 DO 45 I=1,7                                                        TYPE     39
      IF(NEXT(JPTR) .NE. IALPH4(I)) GO TO 110                            TYPE     40
   45 CONTINUE                                                           TYPE     41
      MUL=2                                                              TYPE     42
      ISTATE=2                                                           TYPE     43
      GO TO 60                                                           TYPE     44
C** LOGICAL STATEMENT                                                    TYPE     45
   50 DO 55 I=1,7                                                        TYPE     46
      IF(NEXT(JPTR) .NE. IALPH1(I)) GO TO 110                            TYPE     47
   55 CONTINUE                                                           TYPE     48
      ISTATE=5                                                           TYPE     49
   60 ISUB=0                                                             TYPE     50
      INCR=MUL                                                           TYPE     51
C** GET NEXT VARIABLE IN TYPE STATEMENT                                  TYPE     52
      CALL GNLE                                                          TYPE     53
      IF(JTYP .NE. 2) GO TO 110                                          TYPE     54
      CALL SEARCH                                                        TYPE     55
      IF(ISRCH(2) .EQ. 1) CALL ERROR(10,NXTID)                           TYPE     56
```

```
      IF(ISRCH(1) .EQ. 1) GO TO 62                                      TYPE      57
C** STORE IN SYMBOL TABLE IF NOT FOUND                                  TYPE      58
      IDTYP=1                                                           TYPE      59
      CALL STORE                                                        TYPE      60
      LOC=NID                                                           TYPE      61
C** IF PREVIOUSLY TYPED, ISSUE DIAGNOSTIC                               TYPE      62
   62 IF(BITGET(IDTBL(3,LOC),11,1) .NE. 0) GO TO 120                    TYPE      63
      IF(NEXT(JPTR) .NE. LPAR) GO TO 87                                 TYPE      64
C** DIMENSIONED VARIABLE                                                TYPE      65
      ISUB=1                                                            TYPE      66
      IE=LOC                                                            TYPE      67
      I=0                                                               TYPE      68
   68 I=I+1                                                             TYPE      69
C** GET NEXT DIMENSION                                                  TYPE      70
      CALL GNLE                                                         TYPE      71
      IF(JTYP .NE. 5) GO TO 65                                          TYPE      72
C** DIMENSION IS A CONSTANT, CHECK SIZE                                 TYPE      73
      IDIM(I)=N2                                                        TYPE      74
      IF(N2 .GT. (2**17-1) .OR. N2 .LE. 0) CALL ERROR(8)                TYPE      75
      INCR=INCR*N2                                                      TYPE      76
      GO TO 67                                                          TYPE      77
   65 IF(JTYP .NE. 2) GO TO 110                                         TYPE      78
C** VARIABLE DIMENSION, STORE IN SYMBOL TABLE AND CHECK VALIDITY        TYPE      79
      CALL SEARCH                                                       TYPE      80
      IF(ISRCH(2) .EQ. 1) CALL ERROR(10,NXTID)                          TYPE      81
      IF(ISRCH(1) .EQ. 1) GO TO 66                                      TYPE      82
      IDTYP=1                                                           TYPE      83
      CALL STORE                                                        TYPE      84
      LOC=NID                                                           TYPE      85
   66 IF(BITGET(IDTBL(3,LOC),12,1) .NE. 1) CALL ERROR(9)                TYPE      86
      IF(BITGET(IDTBL(3,LOC),1,1) .NE. 0) GO TO 130                     TYPE      87
C** SET TYPE AND MAKE SURE IT IS INTEGER                                TYPE      88
      CALL IMPTYP                                                       TYPE      89
      IF(BITGET(IDTBL(3,LOC),10,3) .NE. 4) CALL ERROR(9)                TYPE      90
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),1,13)                            TYPE      91
      IDIM(I)=2**17+LOC                                                 TYPE      92
   67 IF(NEXT(JPTR) .EQ. COMMA) GO TO 68                                TYPE      93
      IF(A(JPTR-1) .NE. RPAR) GO TO 110                                 TYPE      94
      K=NEXT(JPTR)                                                      TYPE      95
      LOC=IE                                                            TYPE      96
C** CHECK THAT VARIABLE IS NOT PREVIOUSLY DIMENSIONED                   TYPE      97
C** SET "DIMENSIONED" FLAG                                              TYPE      98
      IF(BITGET(IDTBL(3,LOC),1,1) .NE. 0) CALL ERROR(11,IDTBL(1,LOC))   TYPE      99
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),1,1)                             TYPE     100
      IF(I .GT. 3) GO TO 110                                            TYPE     101
      IF(I-2) 85,80,75                                                  TYPE     102
C** STORE NO. OF DIMENSIONS AND DIMENSION SIZES IN SYMBOL TABLE         TYPE     103
   75 IDTBL(4,LOC)=BITPUT(IDTBL(4,LOC),IDIM(3),36)                      TYPE     104
   80 IDTBL(4,LOC)=BITPUT(IDTBL(4,LOC),IDIM(2),18)                      TYPE     105
   85 IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),IDIM(1),36)                      TYPE     106
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),I,7)                             TYPE     107
   87 IF(INCR .EQ. 1) GO TO 90                                          TYPE     108
      IF(BITGET(IDTBL(3,LOC),16,1) .NE. 1) GO TO 90                     TYPE     109
C** VARIABLE IN COMMON, RESET COMMON BLOCK SIZE IF NECESSARY            TYPE     110
      COMLOC=IDTBL(6,LOC)                                               TYPE     111
      IDTBL(4,COMLOC)=IDTBL(4,COMLOC)+INCR-1                            TYPE     112
C** SET "TYPE SET" FLAG AND STORE TYPE IN SYMBOL TABLE                  TYPE     113
   90 IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),ISTATE,10)                       TYPE     114
      IDTBL(3,LOC)=BITPUT(IDTBL(3,LOC),1,11)                            TYPE     115
      IF(A(JPTR-1) .EQ. COMMA) GO TO 60                                 TYPE     116
      IF(NEXT(JPTR) .EQ. BLANK) RETURN                                  TYPE     117
  110 CALL ERROR(7)                                                     TYPE     118
      RETURN                                                            TYPE     119
  120 CALL ERROR(12,NXTID)                                              TYPE     120
      RETURN                                                            TYPE     121
  130 CALL ERROR(14,NXTID)                                              TYPE     122
      RETURN                                                            TYPE     123
      END                                                               TYPE     124
```

Auxiliary Programs and Associated Data

Program GRAPH

```
      PROGRAM GRAPH(INPUT,TAPE4)
      DIMENSION INT(1000)
   10 FORMAT(12I6)
   20 FORMAT(2I6,A6,I6,A6,7I6/A6,2I6,A6,3I6,A6,I6,A6,2I6/A6,4I6,A6,3I6,
     $ A6,2I6/A6,2I6,A6,8I6/A6,I6,A6,I6,A6,I6,A6,3I6,2A6/3I6,A6,3I6,A6,
     $ 3I6,A6/7I6,A6,4I6/5I6,A6,6I6/(12I6))
      READ 10,(INT(I),I=1,11)
      WRITE(4) (INT(I),I=1,11)
      READ 20,INT
      WRITE(4) INT
      READ 10,INT
      WRITE(4) INT
      READ 10,INT
      WRITE(4) INT
      READ 10,INT
      WRITE(4) INT
      READ 10,INT(1)
      WRITE(4) INT(1)
      READ 10,(INT(I),I=1,34)
      WRITE(4) (INT(I),I=1,34)
      STOP
      END
```

Syntax Graph

1000	99	17	8	17	0	21	74	0	0	1	
359	60'		1-		2	103	103	67	3	3	4
\	72	5RL		0	669	483		10CP		10000	349
•	7	480	795	7950P		20000	21	81		9	42
	18	480'		930	10	935	6	183	935	480	256
(	120		235		13"		14	346	47(	OR	
15	795	795>		16	391	411NT		17	322	359X	
19	28	47	359	29	935	977>		110	67	318	318
29	228	359	40	935/		130	3	72	11	72	60
29	60	999	99	100	101	102	103	104	105	106	40
107	62	109	111	112	113	114	115	116	117	118	119
10	120	122	123	124	125	126	127	81	62	128	21
131	133	134	135	136	137	157	138	93	140	142	143
144	145	146	147	81	480	148	151	152	153	154	155
156	110	157	159	93	114	160	163	164	165	166	167
168	285	60	169	172	173	86	174	88	130	176	179
135	114	72	180	184	185	186	187	188	189	516	190
192	193	194	195	86	130	196	199	133	200	202	203
204	205	206	207	208	209	210	211	212	103	126	208
213	217	208	218	220	221	222	223	224	710	225	553
227	182	229	231	480	292	232	235	126	236	150	238
566	240	242	243	244	157	245	247	248	182	249	251
252	253	254	255	256	257	258	318	150	259	262	175
263	591	265	593	267	269	270	271	272	273	274	228
275	277	278	279	280	281	282	283	284	175	285	287
288	289	290	291	292	293	294	228	295	297	298	299
300	301	302	256	630	303	306	307	378	308	263	310
312	313	314	315	316	317	208	480	318	321	322	346
323	325	326	327	328	329	263	330	332	333	334	288
335	337	338	339	340	341	342	343	670	235	344	347
348	349	350	351	352	353	354	288	355	357	358	418
359	361	315	362	364	365	366	367	368	322	369	371
372	285	373	375	376	377	378	379	380	381	315	382
384	711	385	387	388	322	389	391	392	1	393	285
395	397	398	399	353	400	292	402	404	405	406	407

408	409	410	411	412	346	413	415	416	417	418	419
353	420	422	423	424	425	426	753	318	427	430	431
385	432	86	434	436	437	438	439	440	441	442	443
444	445	446	447	896	448	450	451	385	452	454	455
346	456	110	458	460	461	462	463	464	465	378	466
468	359	796	469	472	473	474	475	476	477	411	478
480	133	481	483	484	485	486	487	378	488	490	491
445	492	494	495	555	496	498	499	500	501	502	503
504	505	506	507	508	509	510	511	445	512	840	514
516	517	518	519	520	521	522	523	524	525	519	526
418	528	483	530	532	533	487	534	536	537	538	539
452	540	542	28	543	545	546	547	548	549	483	550
552	553	487	445	554	557	558	885	559	561	452	562
564	565	566	567	568	569	570	571	572	573	574	60
529	575	578	579	580	581	582	583	584	585	586	72
587	589	480	590	592	483	593	595	529	596	208	598
600	601	602	88	603	931	605	519	607	609	610	611
612	613	614	615	942	616	618	619	620	621	622	623
624	625	626	627	628	629	630	631	632	633	634	635
636	527	637	639	640	641	642	643	644	645	646	647
648	649	650	651	978	652	654	655	982	656	658	659
660	661	662	663	664	665	666	667	668	669	322	670
672	673	674	675	676	677	678	679	680	681	8	682
684	11	685	687	688	689	690	603	691	693	694	695
696	697	698	699	700	701	702	703	704	705	706	707
708	157	709	711	712	603	713	715	669	716	718	719
720	721	722	723	724	725	726	727	728	729	730	731
732	733	734	735	669	736	738	739	740	741	742	743
744	745	746	747	748	749	750	751	752	753	754	755
756	757	758	759	760	761	762	763	764	765	766	767
768	769	770	771	772	773	774	775	776	777	778	779
780	781	108	782	784	785	786	787	788	789	790	791
792	445	793	795	796	797	710	752	452	798	802	803
804	805	806	807	808	809	810	811	812	813	814	815
816	817	818	752	819	821	822	823	824	235	825	827
828	829	830	483	831	318	833	835	836	837	838	839
840	841	842	843	844	845	846	847	848	839	849	851
852	853	854	855	856	857	858	859	860	861	862	863
864	865	866	867	868	869	870	871	872	873	874	875
876	877	878	879	880	881	882	883	884	885	886	887
888	889	890	891	892	893	894	895	896	897	898	899
900	901	902	903	904	795	905	907	908	519	909	911
912	913	914	915	916	917	918	919	920	921	922	923
924	884	925	927	928	929	930	931	932	933	934	935
936	937	938	939	930	940	942	943	944	935	945	947
948	839	949	951	952	953	954	955	956	957	958	959
960	961	962	963	964	965	966	967	968	28	969	971
972	973	974	975	976	889	977	979	980	981	982	983
896	984	986	987	988	989	990	991	992	993	994	995
996	997	998	889								
43	89	4	683	6	66	8	149	710	657	21	24
12	89	617	17	268	19	800	44	28	23	783	13
26	228	130	29	820	31	241	89	266	35	305	37
38	36	130	15	4	345	129	191	89	4	130	178
50	428	94	89	54	471	56	515	445	67	42	61
560	710	839	65	606	5	346	69	653	12	896	73
686	15	4	78	93	535	10	33	889	15	84	555
86	577	89	87	88	10	8	89	597	386	93	98
519	108	100	101	102	103	104	105	106	107	109	96
111	93	112	113	114	115	116	117	118	119	120	122
4	123	124	125	126	127	128	131	1	175	133	4
134	135	136	137	138	140	182	142	133	143	144	145
146	147	148	151	7	130	152	153	154	155	156	157
159	208	160	163	157	228	164	165	166	167	168	169
172	315	5	173	174	176	89	179	89	29	180	184
208	555	11	185	186	187	188	189	190	192	28	193

194	195	196	199	10	603	200	202	235	203	204	205
206	207	208	209	210	211	212	213	217	5	89	110
218	220	452	221	222	223	224	225	227	483	229	25
231	133	232	235	130	235	236	238	3	240	157	242
30	243	244	245	247	89	248	249	251	150	252	253
254	255	256	257	258	259	262	89	3	263	265	519
267	80	269	16	270	271	272	273	274	275	277	256
278	279	280	281	282	283	284	285	287	3	288	289
290	291	292	293	294	295	297	285	298	299	300	301
302	303	306	346	34	307	308	310	411	312	29	313
314	315	316	317	318	321	1	130	322	323	325	445
326	327	328	329	330	332	292	333	334	335	337	263
338	339	340	341	342	343	344	347	59	2	348	349
350	351	352	353	354	355	357	889	358	359	361	89
362	364	256	365	366	367	368	369	371	208	372	373
375	89	376	377	378	379	380	381	382	384	285	385
387	51	388	389	391	126	392	393	395	21	397	13
398	399	400	402	208	404	1	405	406	407	408	409
410	411	412	413	415	89	416	417	418	419	420	422
89	423	424	425	426	427	430	49	5	431	432	434
29	436	529	437	438	439	440	441	442	443	444	445
446	447	448	450	935	451	452	454	418	455	456	458
13	460	288	461	462	463	464	465	466	468	89	469
472	4	53	473	474	475	476	477	478	480	378	481
483	263	484	485	486	487	488	490	13	491	492	494
29	495	496	498	603	499	500	501	502	503	504	505
506	507	508	509	510	511	512	514	480	516	55	517
518	519	520	521	522	523	524	525	526	528	62	530
3	532	669	533	534	536	76	537	538	539	540	542
89	543	545	47	546	547	548	549	550	552	710	553
554	557	83	6	558	559	561	60	562	564	5	565
566	567	568	569	570	571	572	573	574	575	578	81
85	579	580	581	582	583	584	585	586	587	589	529
590	592	7	593	595	14	596	598	95	600	110	601
602	603	605	669	607	64	609	89	610	611	612	613
614	615	616	618	40	619	620	621	622	623	624	625
626	627	628	629	630	631	632	633	634	635	636	637
639	5	640	641	642	643	644	645	646	647	648	649
650	651	652	654	68	655	656	658	90	659	660	661
662	663	664	665	666	667	668	669	670	672	385	673
674	675	676	677	678	679	680	681	682	684	3	685
687	72	688	689	690	691	693	89	694	695	696	697
698	699	700	701	702	703	704	705	706	707	708	709
711	182	712	713	715	9	716	718	208	719	720	721
722	723	724	725	726	727	728	729	730	731	732	733
734	735	736	738	88	739	740	741	742	743	744	745
746	747	748	749	750	751	752	753	754	755	756	757
758	759	760	761	762	763	764	765	766	767	768	769
770	771	772	773	774	775	776	777	778	779	780	781
782	784	22	785	786	787	788	789	790	791	792	793
795	114	796	797	798	802	89	18	487	803	804	805
806	807	808	809	810	811	812	813	814	815	816	817
818	819	821	28	822	823	824	825	827	322	828	829
830	831	833	752	835	40	836	837	838	839	840	841
842	843	844	845	846	847	848	849	851	884	852	853
854	855	856	857	858	859	860	861	862	863	864	865
866	867	868	869	870	871	872	873	874	875	876	877
878	879	880	881	882	883	884	885	886	887	888	889
890	891	892	893	894	895	896	897	898	899	900	901
902	903	904	905	907	15	908	909	911	62	912	913
914	915	916	917	918	919	920	921	922	923	924	925
927	930	928	929	930	931	932	933	934	935	936	937
938	939	940	942	977	943	944	945	947	3	948	949
951	4	952	953	954	955	956	957	958	959	960	961
962	963	964	965	966	967	968	969	971	710	972	973
974	975	976	977	979	89	980	981	982	983	984	986

99	987	988	989	990	991	992	993	994	995	996	997
998	999	99	3								
43	2	4	3	6	5	8	7	9	90	11	13
12	14	40	17	16	19	18	44	21	23	22	24
26	25	110	29	28	31	30	32	80	35	34	717
38	37	191	15	41	59	1	20	45	46	276	52
50	49	94	324	54	53	56	55	394	58	42	61
60	737	311	65	64	66	459	69	68	433	370	73
72	139	158	78	77	76	493	33	82	178	84	83
86	85	89	88	87	10	241	92	95	51	93	98
97	96	0	0	0	0	0	0	0	0	0	108
0	27	0	0	0	0	0	0	0	0	0	0
121	0	0	0	0	0	0	0	129	130	0	132
0	0	0	0	0	0	74	0	141	0	0	0
0	0	0	0	149	150	0	0	0	0	0	0
0	75	0	0	161	162	0	0	0	0	0	0
0	170	171	0	0	0	175	0	177	81	0	0
181	182	183	0	0	0	0	0	0	0	39	0
0	0	0	0	197	198	0	0	201	0	0	0
0	0	0	0	0	0	0	0	0	214	215	216
0	0	219	0	0	0	0	0	0	226	0	228
0	230	0	0	233	234	0	0	237	0	239	0
91	0	0	0	0	246	0	0	0	250	0	0
0	0	0	0	0	0	0	260	261	0	0	264
0	266	0	268	0	0	0	0	0	0	0	47
0	0	0	0	0	0	0	0	0	286	0	0
0	0	0	0	0	0	0	296	0	0	0	0
0	0	0	304	305	0	0	0	309	0	63	0
0	0	0	0	0	0	319	320	0	0	0	48
0	0	0	0	0	0	331	0	0	0	0	336
0	0	0	0	0	0	0	0	345	346	0	0
0	0	0	0	0	0	0	356	0	0	0	360
0	0	363	0	0	0	0	0	0	71	0	0
0	374	0	0	0	0	0	0	0	0	383	0
0	386	0	0	0	390	0	0	0	57	0	396
0	0	0	0	401	0	403	0	0	0	0	0
0	0	0	0	0	414	0	0	0	0	0	0
421	0	0	0	0	0	0	428	429	0	0	0
70	0	435	0	0	0	0	0	0	0	0	0
0	0	0	0	449	0	0	0	453	0	0	0
457	0	67	0	0	0	0	0	0	0	467	0
0	470	471	0	0	0	0	0	0	0	479	0
0	482	0	0	0	0	0	0	489	0	0	0
79	0	0	0	497	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	513	0	515	0
0	0	0	0	0	0	0	0	0	0	527	0
529	0	531	0	0	0	535	0	0	0	0	0
541	0	0	544	0	0	0	0	0	0	551	0
0	0	555	556	0	0	0	560	0	0	563	0
0	0	0	0	0	0	0	0	0	0	0	576
577	0	0	0	0	0	0	0	0	0	0	588
0	0	591	0	0	594	0	0	597	0	599	0
0	0	0	604	0	606	0	608	0	0	0	0
0	0	0	0	617	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	638	0	0	0	0	0	0	0	0	0	0
0	0	0	0	653	0	0	0	657	0	0	0
0	0	0	0	0	0	0	0	0	0	671	0
0	0	0	0	0	0	0	0	0	0	683	0
0	686	0	0	0	0	0	692	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	710	0	0	0	714	0	0	36	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	62	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0

0	0	0	0	0	0	0	0	0	0	0	0
1	0	783	0	0	0	0	0	0	0	0	0
0	794	0	0	0	0	799	800	801	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	820	0	0	0	0	0	826	0	0
0	0	0	832	0	834	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	850	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	906	0	0	0	910	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	926	0	0	0	0	0	0	0	0	0	0
0	0	0	0	941	0	0	0	0	946	0	0
0	950	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	970	0	0
0	0	0	0	0	978	0	0	0	0	0	0
985	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1000								
96	76	99	97	99	97	96	96	76	97	76	97
99	76	97	99	97	96	96	99	76	99	97	236
99	97	72	96	96	99	97	76	97	99	97	232
99	97	72	99	77	97	96	97	76	77	72	72
99	97	99	72	99	97	99	97	72	76	99	99
97	72	72	99	97	236	72	99	97	73	72	99
97	73	73	96	76	96	72	99	72	73	96	96
96	96	96	236	96	99	73	76	96	97	96	96
76	96	0	0	0	0	0	0	0	0	0	236
0	76	0	0	0	0	0	0	0	0	0	0
77	0	0	0	0	0	0	0	236	76	0	77
0	0	0	0	0	0	76	0	76	0	0	0
0	0	0	0	236	76	0	0	0	0	0	0
0	76	0	0	76	76	0	0	0	0	0	0
0	76	77	0	0	0	76	0	76	76	0	0
76	76	77	0	0	0	0	0	0	0	236	0
0	0	0	0	77	76	0	0	76	0	0	0
0	0	0	0	0	0	0	0	0	77	76	76
0	0	76	0	0	0	0	0	0	76	0	236
0	76	0	0	76	76	0	0	77	0	76	0
236	0	0	0	0	76	0	0	0	76	0	0
0	0	0	0	0	0	0	76	77	0	0	76
0	236	0	236	0	0	0	0	0	0	0	76
0	0	0	0	0	0	0	0	0	77	0	0
0	0	0	0	0	0	0	76	0	0	0	0
0	0	0	76	236	0	0	0	76	0	76	0
0	0	0	0	0	0	77	76	0	0	0	76
0	0	0	0	0	0	76	0	0	0	0	76
0	0	0	0	0	0	0	0	236	77	0	0
0	0	0	0	0	0	0	76	0	0	0	76
0	0	76	0	0	0	0	0	0	76	0	0
0	76	0	0	0	0	0	0	0	0	76	0
0	236	0	0	0	76	0	0	0	76	0	77
0	0	0	0	76	0	77	0	0	0	0	0
0	0	0	0	0	76	0	0	0	0	0	0
76	0	0	0	0	0	0	236	77	0	0	0
76	0	76	0	0	0	0	0	0	0	0	0
0	0	0	0	76	0	0	0	76	0	0	0
77	0	76	0	0	0	0	0	0	0	76	0
0	77	236	0	0	0	0	0	0	0	76	0
0	76	0	0	0	0	0	0	77	0	0	0
76	0	0	0	76	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	76	0	236	0
0	0	0	0	0	0	0	0	0	0	76	0
77	0	76	0	0	0	236	0	0	0	0	0
76	0	0	76	0	0	0	0	0	0	76	0

0	0	236	77	0	0	0	236	0	0	77	0
0	0	0	0	0	0	0	0	0	0	0	76
236	0	0	0	0	0	0	0	0	0	0	76
0	0	77	0	0	77	0	0	236	0	76	0
0	0	0	76	0	236	0	76	0	0	0	0
0	0	0	0	236	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	77	0	0	0	0	0	0	0	0	0	0
0	0	0	0	236	0	0	0	236	0	0	0
0	0	0	0	0	0	0	0	0	0	76	0
0	0	0	0	0	0	0	0	0	0	236	0
0	236	0	0	0	0	0	76	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	76	0	0	0	77	0	0	76	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	76	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	236	0	0	0	0	0	0	0	0	0
0	76	0	0	0	0	76	236	76	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	236	0	0	0	0	0	76	0	0
0	0	0	76	0	76	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	76	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	77	0	0	0	76	0	0
0	0	0	0	0	0	0	0	0	0	0	0
0	76	0	0	0	0	0	0	0	0	0	0
0	0	0	0	76	0	0	0	0	76	0	0
0	77	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	76	0	0
0	0	0	0	0	76	0	0	0	0	0	0
76	0	0	0	0	0	0	0	0	0	0	0
0	0	0	77								
0											
8	391	982	349	942	516	553	591	630	670	711	753
796	840	885	931	978	42	11	48	68	89	111	675
154	642	137	593	108	566	19	29	228	483		

Program SESLIST

```
      PROGRAM SESLIST(INPUT,OUTPUT,TAPE5=INPUT,LIST,TAPE4=LIST)
      INTEGER TYPE,CLASS,BLKTYP(100),BLKSIZ(100)
      DIMENSION ISUBLT(2,200),INTFAC(300),IARGS(200)
      EQUIVALENCE (BLKTYP(1),IARGS(1)),(BLKSIZ(1),IARGS(101))
      NLIST=0
      IPTR=1
    1 READ(5,5) NAME,NARGS,TYPE,CLASS,ISIZE
    5 FORMAT(A6,1X,3I2,1X,I6)
      IF(ISIZE .LT. 1) ISIZE=0
      IF(EOF(5) .NE. 0) GO TO 40
      NLIST=NLIST+1
      IVAR=0
      ISUBLT(1,NLIST)=NAME
      IF(NARGS .EQ. -1) NARGS=IVAR=1
      ISUBLT(2,NLIST)=SHIFT(NARGS,54) .OR. SHIFT(TYPE,47) .OR.
     $ SHIFT(CLASS,50) .OR. SHIFT(ISIZE,30)
      IF(NARGS .EQ. 0) GO TO 1
      ISUBLT(2,NLIST)=ISUBLT(2,NLIST) .OR. IPTR .OR. SHIFT(IVAR,46)
      IF(CLASS .EQ. 7) GO TO 25
      JPTR=IPTR
      INC=1+(NARGS-1)/6
      IPTR=IPTR+INC
      NDPTR=IPTR-1
      NPARAM=3*NARGS
      READ(5,10) (IARGS(I),I=1,NPARAM)
   10 FORMAT(20(3I1,1X))
      KOUNT=0
      DO 20 I=JPTR,NDPTR
      INTFAC(I)=0
      NSHIFT=60
      DO 20 K=1,18
      KOUNT=KOUNT+1
      IF(KOUNT .GT. NPARAM) GO TO 1
      NSHIFT=NSHIFT-3
   20 INTFAC(I)=INTFAC(I) .OR. SHIFT(IARGS(KOUNT),NSHIFT)
      GO TO 1
   25 JPTR=IPTR
      INC=1+(NARGS-1)/3
      IPTR=IPTR+INC
      NDPTR=IPTR-1
      READ(5,27) (BLKSIZ(I),BLKTYP(I),I=1,NARGS)
   27 FORMAT(10(I6,1X,I1))
      KOUNT=0
      DO 30 I=JPTR,NDPTR
      INTFAC(I)=0
      NSHIFT=60
      DO 30 K=1,3
      KOUNT=KOUNT+1
      IF(KOUNT .GT. NARGS) GO TO 1
      NSHIFT=NSHIFT-17
      INTFAC(I)=INTFAC(I) .OR. SHIFT(BLKSIZ(KOUNT),NSHIFT)
      NSHIFT=NSHIFT-3
   30 INTFAC(I)=INTFAC(I) .OR. SHIFT(BLKTYP(KOUNT),NSHIFT)
      GO TO 1
   40 WRITE(4) NLIST,NDPTR
      WRITE(4) ((ISUBLT(I,J),I=1,2),J=1,NLIST)
      WRITE(4) (INTFAC(I),I=1,NDPTR)
      PRINT 50,NLIST
   50 FORMAT(////42X,*NEW LIST HAS BEEN CREATED CONTAINING*,I4,* NAMES*)
      STOP
      END
```

Basic Interface Definition File

```
ABS      1 1 4
102
AINT     1 1 4
102
ALOG     1 1 4
102
ALOG10   1 1 4
102
AMAX0   -1 1 4
402
AMAX1   -1 1 4
102
AMIN0   -1 1 4
402
AMIN1   -1 1 4
102
AMOD     2 1 4
102 102
AIMAG    1 1 4
202
ATAN     1 1 4
102
ATAN2    2 1 4
102 102
CABS     1 1 4
202
CCOS     1 2 4
202
CEXP     1 2 4
202
CLOG     1 2 4
202
CMPLX    2 2 4
102 102
CONJG    1 2 4
202
COS      1 1 4
102
CSIN     1 2 4
202
CSQRT    1 2 4
202
DABS     1 3 4
302
DATAN    1 3 4
302
DATAN2   2 3 4
302 302
DBLE     1 3 4
102
DCOS     1 3 4
302
DEXP     1 3 4
302
DIM      2 1 4
102 102
DLOG     1 3 4
302
DLOG10   1 3 4
302
DMAX1   -1 3 4
302
DMIN1   -1 3 4
302
DMOD     2 3 4
302 302
DSIGN    2 3 4
302 302
DSIN     1 3 4
302
DSQRT    1 3 4
302
EXP      1 1 4
102
FLOAT    1 1 4
402
IABS     1 4 4
402
IDIM     2 4 4
402 402
IDINT    1 4 4
302
IFIX     1 4 4
102
INT      1 4 4
102
ISIGN    2 4 4
402 402
MAX0    -1 4 4
402
MAX1    -1 4 4
102
MIN0    -1 4 4
402
MIN1    -1 4 4
102
MOD      2 4 4
402 402
REAL     1 1 4
202
SESCOM   2 0 7      25
    13 0     12 4
SIGN     2 1 4
102 102
SIN      1 1 4
102
SNGL     1 1 4
302
SQRT     1 1 4
102
TAN      1 1 4
102
TANH     1 1 4
102
```

INITIAL DISTRIBUTION

Copies	
10	NAVSEA PMS304-32 White
2	NAVSEA PMS405-40 Cuthbert
12	DDC

CENTER DISTRIBUTION

1	18/1809
1	1802.2 Frenkiel
1	1802.4 Theilheimer
1	1809.3 D. Harris (Central Depository, CMLD)
1	182 Camara
1	1826 Culpepper
30	1826 Wybraniec
1	184 Lugt
1	185 Corin
1	186 Sulit
1	189 Gray
1	1890 Taylor
30	5214.1 Reports Distribution
1	522

Microfiche copies

30	1826 Wybraniec

DTNSRDC ISSUES THREE TYPES OF REPORTS

(1) DTNSRDC REPORTS, A FORMAL SERIES PUBLISHING INFORMATION OF PERMANENT TECHNICAL VALUE, DESIGNATED BY A SERIAL REPORT NUMBER.

(2) DEPARTMENTAL *REPORTS, A SEMIFORMAL SERIES,* RECORDING INFORMATION OF A PRELIMINARY OR TEMPORARY NATURE, OR OF LIMITED INTEREST OR SIGNIFICANCE, CARRYING A DEPARTMENTAL ALPHANUMERIC IDENTIFICATION.

(3) TECHNICAL MEMORANDA, AN INFORMAL SERIES, USUALLY INTERNAL WORKING PAPERS OR DIRECT REPORTS TO SPONSORS, NUMBERED AS TM SERIES REPORTS; NOT FOR GENERAL DISTRIBUTION.